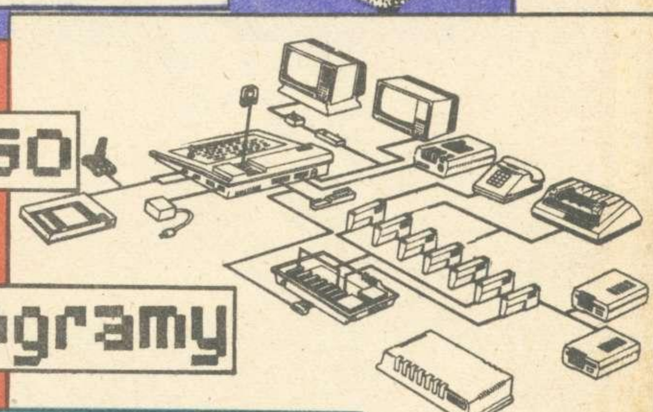
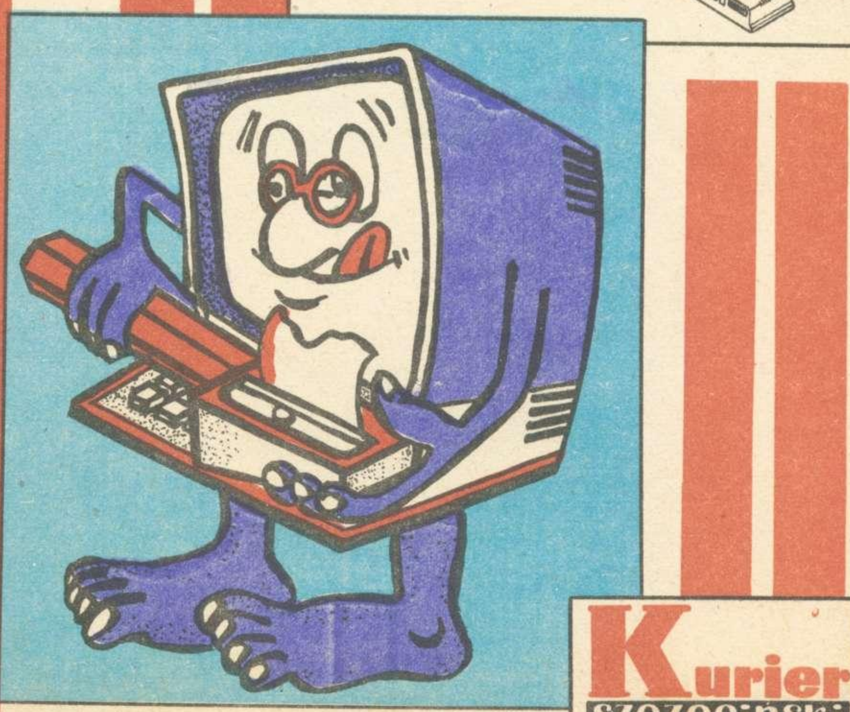
MIKROKOMPUTERY

abc ABC abc



- Podstawy BASICA
- Polskie LOGO
- Krótkie programy





Kurier szczeciński

MIKROKOMPUTERY

Autorzy:

Piotr Buczyński
Mirosław Frąckiewicz
Piotr Kuczera
Antoni Nowakowski
Jacek Szydłowski

Szczecin 1987

Drodzy Czytelnicy!

ODDAJEMY do waszych rąk wydawnictwo poświęcone mikrokomputerom. O znaczeniu komputerów we współczesnym świecie już nikogo nie trzeba przekonywać. Od niedawna zadomowiły się one już na stałe w wielu przedsiębiorstwach i instytucjach, w szkołach i na wyższych uczelniach. W wielu domach stały się one przyjacielem i doradcą.

W Polsce mamy za sobą pierwszy etap fascynacji możliwościami popularnych mikrokomputerów, sprowadzający się do stosowania ich wyłącznie do mniej lub bardziej sensownych gier. Ważne jest, że komputer staje się w szkołach przyjacielem młodzieży na lekcjach jako pomoc dydaktyczna a także w licznych pracowniach i klubach komputerowych urządzeniem do wstępnej nauki programowania.

Proces alfabetyzacji komputerowej rozszerza się coraz bardziej. W wielu szkołach średnich organizowane są już zajęcia z podstaw informatyki. Wiele instytucji organizuje kursy dla pracowników przedsiębiorstw przemysłowych i instytucji. Komputery stosowane są już jako nieodłączne narzędzie pracy nie tylko pracowników służb ekonomicznych, ale także inżynierów i projektantów.

Już niedługo nadejdą czasy, że bez abecadła informatycznego współczesny człowiek będzie miał poważne trudności w poruszaniu się we współczesnym świecie. Postanowiliśmy i my wyjść tym zainteresowaniom i potrzebom naprzeciw, wydając krótkie kompendium wiedzy o mikrokomputerach.

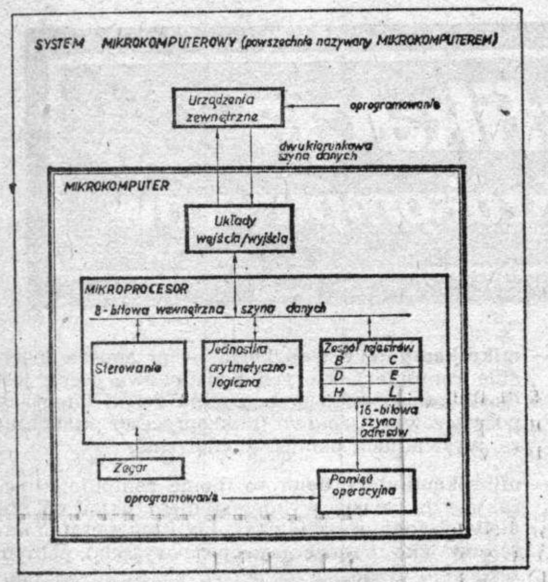
Od lutego 1986 roku dwa razy w miesiącu w „Mikrokurierze" stałym dodatku do „Kuriera Szczecińskiego" prezentujemy wiadomości z zakresu informatyki i mikrokomputerów, przeznaczone dla młodzieży. Kilku współpracujących z nami autorów i animatorów ruchu mikrokomputerowego w naszym województwie, pracowników naukowych Instytutu Cybernetyki Ekonomicznej i Informatyki Uniwersytetu Szczecińskiego wzięło udział w opracowaniu wydawnictwa, które trzymacie w ręku. Określone trudności związane z papierem, kłopoty z bazą poligraficzną wpłynęły na to, że cykl wydawniczy niniejszej publikacji nieco się wydłużył i wydawnictwo dociera do was później niż zakładali to jego autorzy. Wybaczcie nam jednak, znając uwarunkowania, w jakich ono się ukazuje. Staramy się za to zrekompensować je stosunkowo dostępną ceną.

Zapraszając was do przeczytania naszego wydawnictwa liczymy się z tym, że pozwoli ono na uporządkowanie posiadanej wiedzy, wzbogacenie jej o nieznane dotąd treści a tych, dla których stanowi ono pierwszy kontakt z wiedzą o mikrokomputerach zachęci do zainteresowania się informatyką. Cieszylibyśmy się bardzo, gdyby dzięki naszej publikacji grono zwolenników stosowania mikrokomputerów stało się jeszcze większe.

SPIS TREŚCI

1 Budowa mikrokomputerów

MIKROKOMPUTER zbudowany jest z trzech podstawowych elementów:
- **mikroprocesora z układem synchronizacji wewnętrznej,**
- **pamięci operacyjnej (wewnętrznej),**
- **układów służących do łączenia mikroprocesora z urządzeniami wejścia-wyjścia, podłączonych za pomocą szyn przesyłania informacji (adresów, danych i sygnałów sterujących). Jego ogólną strukturę przedstawia rys. 1.1.**



RYS. 1.1 STRUKTURA SYSTEMU MIKROKOMPUTEROWEGO

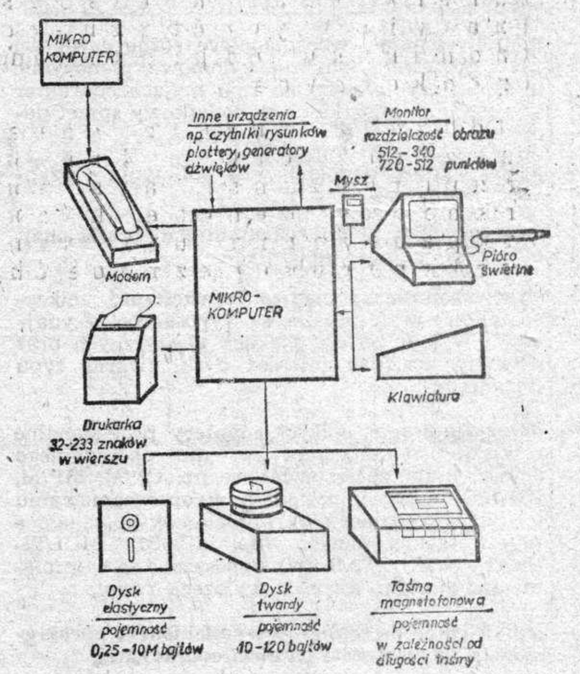
PODSTAWOWYM elementem mikrokomputera jest **mikroprocesor**. Jest to układ scalony o bardzo dużym stopniu integracji (VLSI) zawierający od kilkunastu do kilkuset tysięcy tranzystorów umieszczonych na niewielkiej powierzchni i stanowiących jeden element (całość), a komunikujących się z otoczeniem poprzez określoną liczbę wyprowadzeń (najczęściej 40 dla mikroprocesora 8-bitowego). Posiada on zdolność wykonywania ciągu rozkazów.

Mikroprocesor koordynuje wszelką działalność systemu, a przede wszystkim pobiera, dekoduje i wykonuje rozkazy umieszczone w pamięci. Zegar połączony z mikroprocesorem wytwarza ciąg sygnałów wykorzystywanych do synchronizacji działania mikroprocesora. Typowy mikroprocesor zbudowany jest z czterech bloków funkcjonalnych
- **sterowania** (interpretacja rozkazów pobieranych z pamięci, a następnie kontrolowanie na tej podstawie wewnętrznego działania mikroprocesora oraz reakcje na sygnały z zewnątrz, jak również wytwarzanie sygnałów umożliwiających organizowanie współpracy innych elementów systemu z mikroprocesorem,
- **jednostki automatyczno-logicznej (ALU), wykonującej obliczenia arytmetyczne i logiczne,**
- **rejestrów** służących do zapisu i odczytu danych,
- **wewnętrznych szyn przesyłowych** wiążących wymienione bloki.

Do najważniejszych parametrów charakteryzujących określony mikroprocesor należą: długość słowa maszynowego, szybkość pracy i liczba rozkazów znajdujących się w repertuarze mikroprocesora.

Pierwszą generację stanowią mikroprocesory 8-bitowe co oznacza, że operuje informacjami (przesyła, przechowuje w rejestrach) 8-bitowymi. Produkcja mikroprocesorów 8-bitowych na szeroką skalę rozpoczęła się w 1972 r. Kolejna generacja to mikroprocesory 16-bitowe (1978 r.), 32-bitowe (1981—1982 r.), a w 1986 roku spodziewana jest czwarta generacja mikroprocesorów (64-bitowych).

Lista rozkazów mikroprocesora zawiera wykaz elementarnych części programów, które mikroprocesor będzie w stanie realizować. Lista ta zawiera od 30 do 150 rozkazów w zależności od przeznaczenia i złożoności mikroprocesora. Dlatego też jej długość nie zawsze oznacza większe bądź mniejsze możliwości. I tak niewielka liczba rozkazów przy szybkich rejestrach mikroprocesora i odpowiednich mechanizmach pobierania następnego rozkazu powoduje wzrost możliwości użytkowych. Rozbudowana natomiast lista rozkazów umożliwia stosunkowo prostą implementację dla określonego mikroprocesora języków wysokiego poziomu. W liście rozkazów odwzorowana jest architektura każdego mikroprocesora. Kod rozkazu jest podstawą dla układu sterowania wytwarzania sygnałów, które wywołują działania różnych układów mikrokomputera.



RYS. 1.2 MOŻLIWOŚCI BUDOWY KONFIGURACJI MIKROKOMPUTERÓW

Pamięć operacyjna służy do przechowywania rozkazów wykonywanych przez mikroprocesor oraz danych, których rozkazy te dotyczą. Pamięć operacyjną

mikrokomputerów przyrównać można do brulionu (brudnopisu) używanego w trakcie wykonywania obliczeń. W brulionie tym mikrokomputer wielokrotnie przepisuje dane a także wyniki pośrednie obliczeń. Porównanie to odnosi się do tej części pamięci operacyjnej, która w mikrokomputerach nosi nazwę RAM (Random Access Memory), a użytkownik systemu może w niej zapisywać informacje. Druga część pamięci ROM (Read Only Memory) zapisywana jest w trakcie produkcji, a system odczytuje tylko z niej informacje, nie może ich zmodyfikować czy wymazać, zapisane w niej informacje nie niszczą się w czasie zaniku zasilania elektrycznego, są zapisane na stałe.

Układy wejścia-wyjścia służą do dołączenia do mikrokomputera różnych typów urządzeń wejścia-wyjścia (urządzeń zewnętrznych). Najczęściej spotykanymi urządzeniami wejścia-wyjścia są monitory ekranowe, pamięci na dyskach (twardych i elastycznych) i kasetach magnetycznych, klawiatury, drukarki oraz łącza umożliwiające łączenie mikrokomputerów w sieci oraz komunikację z innymi komputerami. Zestaw wyprowadzeń do urządzeń zewnętrznych zależy od klasy mikrokomputera, umożliwi tworzenie różnych konfiguracji (zestawów) użytkowych.

Szyny (BUSy) służą do przesyłania informacji pomiędzy wymienionymi elementami tworząc z nich mikrokomputer. Szerokość wewnętrznych (w mikroprocesorze) i zewnętrznych szyn przesyłania (8, 16 i 32-bitowe) decyduje o możliwości adresowania pamięci operacyjnej, a zatem i o wielkości tej pamięci. Przy 16-bitowej szynie adresów można zaadresować 65536 bajtów (=64 K bajtów).

Urządzenia zewnętrzne mikrokomputera (i każdego komputera) umożliwiają:

— komunikację człowieka z mikrokomputerem (klawiatura, monitor ekranowy, drukarka),
— rozszerzenie jego zasobów (pamięci zewnętrzne, podłączenia do innych systemów komputerowych).

Możliwości wyposażenia mikrokomputera w urządzenia zewnętrzne przedstawia rys. 1.2.

2 Klasyfikacja i przegląd mikrokomputerów

PRZYJMUJĄC jako podstawowe kryterium podziału wg adresata zastosowań, mikrokomputery podzielić można na:

— **nieprofesjonalne**, wspomagające różne czynności życia domowego (obliczenia domowe, planowanie budżetu, gry telewizyjne itp.) mające charakter wyposażenia mieszkania w dodatkowy sprzęt domowy,

— **profesjonalne**, wspomagające obliczenia i czynności zawodowe.

Można przyjąć, że mikrokomputer nieprofesjonalny od profesjonalnego odróżnia:

— **stosowany rodzaj pamięci zewnętrznej**, mikrokomputery profesjonalne są wyposażone przynajmniej w pamięci na dyskach elastycznych oraz powinny posiadać również dyski twarde typu Winchester,

— **oprogramowanie**, mikrokomputery profesjonalne pracują pod nadzorem systemu operacyjnego (chodzi o standardy światowe np. CP/M, MP/M, UNIX, MS-DOS), posiadają w oprogramowaniu użytkowym pakiety uniwersalne wspomagające przetwarzanie danych (np. dBASE, MULTIPLAN, WORDSTAR itp.) wyposażone są w implementacje wielu języków wyższego rzędu,

— **odpowiednie do realizacji wymienionego oprogramowania pojemności pamięci operacyjnej,**

— **zestaw urządzeń peryferyjnych** umożliwiających wykorzystanie w pracy zawodowej.

Wśród mikrokomputerów nieprofesjonalnych wyróżnia się zazwyczaj:

— **mikrokomputery kieszonkowe, spełniające praktycznie funkcje kalkulatorów programowanych,**

— **mikrokomputery walizkowe** — przenośne (portable computer), których podstawową cechą jest możliwość łatwego przenoszenia dzięki integracji podstawowego zestawu (mikroprocesor, klawiatura, wyświetlacz, pamięć zewnętrzna),

— **mikrokomputery domowe** (home computer) charakteryzujące się wykorzystaniem urządzeń domowych (odbiornik telewizyjny i magnetofon kasetowy jako urządzenia wejścia-wyjścia), pełnym zestawem urządzeń zewnętrznych, możliwościami realizacji obliczeń zawodowych (ale w domu).

Mikrokomputery profesjonalne można podzielić według kryterium długości słowa na:

— **mikrokomputery profesjonalne 8-bitowe,**
— **mikrokomputery profesjonalne 16-bitowe,**
— **mikrokomputery profesjonalne 32-bitowe.**

W dwóch tablicach 2.1 oraz 2.2 przedstawione zostały wybrane mikrokomputery produkowane w Polsce przez przedsiębiorstwa państwowe i firmy polonijne oraz mikrokomputery zagraniczne popularne w Polsce w zastosowaniach profesjonalnych i domowych.

W tablicach tych przedstawiono tylko niektóre parametry świadczące o zasobach mikrokomputerów, istotnych dla użytkowników. Przyjąć należy, że są to konfiguracje i wielkości standardowe, że można je rozbudowywać, dostosowywać do potrzeb użytkownika.

Tablica 2.1 charakteryzuje mikrokomputery produkcji krajowej z podziałem na profesjonalne i nieprofesjonalne. Znajdujące się w nich parametry wskazują na ich niewielkie zróżnicowanie, wynikające z możliwości tworzenia konfiguracji. Przodują wśród nich ComPAN 8 (którego jednak produkcja jest niewielka, jak również jej perspektywy) i Elwro 800 (który jest jeszcze jednak przedsięwzięciem planowanym). Dostępność określonych drukarek (z reguły DZM 180), monitorów (z reguły Neptun 156),

Tablica 2.1

Przegląd wybranych mikrokomputerów produkowanych w Polsce

Mikrokomputer / Parametry	PROFESJONALNE										NIEPROFESJONALNE		
	ELWRO 523	ELWRO 600	ELWRO 800 założenia	PSPD 90	MK 45	ComPan 8	IMP 85	CS 80	IMZ 80	MERA 80	MERITUM I	ZX SPECTRUM	Unipolbrit 2086
PAMIĘĆ ROM (w KB)	12	8	moduł 16-bitowy 8-64	4	4	32	4	18	2	8-32	14	16	24
PAMIĘĆ RAM (w KB)	48	64	32-128 dodatkowe moduły po 256	12-64	16-64	128-512	64	64	64	16-64	16-48	48	48
PAMIĘĆ ZEWNĘTRZNA — rozmiar (w calach) — ilość dyskietek — pojemność dyskietki (KB)	8 2 256	5 1/4 2 lub 4 75	5 1/4-8 2-4 w zależności od dyskietki	8 4 256	8 2 256	5 1/4-8 4 w zależności od dyskietki	8 2 256	8 2 256	8 2 242	8 2-4 256	dyski elast. 5 1/2 przewidziane są w modelu MERITUM II	poprzez rozbudowę konfiguracji	opcjonalnie
Drukarka — prędkość drukowania (znak/sek) — ilość znaków w wierszu — grafika	40 132-158 —	100 80-132 —	w zależ. od zainstalowanej drukarki +	180 132-158 —	180 132-158 —	180 132-158 +	180 132-158 —	180 132-158 —	280 132-158 —	180 132-158	opcjonalnie	opcjonalnie	opcjonalnie
Monitor — ilość wierszy — ilość znaków w wierszu — rozdzielczość (w punktach)	16 64 480x200	25 80 480x200	25 80 640x200	16 64 480x200	25 80 480x200	25 80 640x200	25 80 480x200	25 80 480x200	25 80 480x200	25 80 480x200	16 64 480x200	24 32 256x176	24 64 512x192

Tablica 2.2

Przegląd wybranych mikrokomputerów zagranicznych

Mputer / ram	Nieprofesjonalne			Przejściowe między nieprofesjonalnymi a profesjonalnymi				Profesjonalne		
						Przenośne				
	Commodore C 64	Amstrab CPC 6128	Atari 800 XL	Sindair QL	Commodore C 128	Commodore LCD	Olivetti M 21	Apple II c	IBM PC	Macintosh
Pamięć ROM (w KB)	20	48	24	48	18	96	32	16	40	64
Pamięć RAM (w KB)	64	128	64	128-640	128-512	32	32-64	128	64-512	128-512
Pamięć zewnętrzna				specjalny micro drive 100 KB						
a) kasety magnetofonowe	+	+	+	—	+	—	—	—	—	—
b) dyski elastyczne						+				
— rozmiar (w calach)	51/4	3	51/4		51/4		3	51/4	51/4	31/2
— ilość dyskietek	2	2	4		2		2	2	2	
— pojemność dyskietki (w KB)	170	180	127		350		320	143	160-320	400
Monitor										
— ilość wierszy	25	25	24	25	25	16	24	24	25	24
— ilość znaków w wierszu	40	40	40	40	80	80	80	40	80	80
— rozdzielczość (w punktach)	320x200	640x200	320x192	512x258	640x200	480x128	640x400	560x192	40x200	512x342
Drukarki	różnorodne, własne konstrukcje producentów mikrokomputerów lub specjalistycznych producentów drukarek									

napędów dysków elastycznych tworzy możliwości budowy konfiguracji i jej zasobów. Parametry większości mikrokomputerów są na poziomie minimum wymaganego przez sprzęt profesjonalny. Większe natomiast zróżnicowanie prezentowanych mikrokomputerów występuje w odniesieniu do takich cech jak:

— dostępność, w zasadzie powszechnie dostępnymi są mikrokomputery ELWRO 523 i produkcji firm polonijnych,
— cena, mikrokomputery przedsiębiorstw państwowych z reguły są droższe, a wśród nich ELWRO 523, którego koszt zakupu i instalacji sięga obecnie około 2,3 mln zł,
— obsługa techniczna, w zasadzie oprócz Elwro 523 (dla którego serwis prowadzi BIUROTECHNIKA) nie ma zorganizowanego systemu napraw, poza naprawami gwarancyjnymi realizowanymi przez producentów.

Te trzy wymienione cechy są jednak bardzo istotne przy podejmowanych przez użytkowników decyzjach zakupu.

W odniesieniu do mikrokomputerów nieprofesjonalnych można stwierdzić, że znajdują one powszechne zastosowania (szczególnie ZX Spectrum, a coraz szerzej MERITUM I w zestawach dydaktycznych z Meritum II) we wszelkiego rodzaju klubach i laboratoriach szkolnych, których celem jest zapoznanie młodzieży z mikrokomputerami (budowa, operatorka, programowanie). Tu następuje połączenie z zastosowaniami domowymi mikrokomputerów. Zainteresowanie działalnością takich laboratoriów wynika bowiem z jednej strony z chęci „opanowania" posiadanego już prywatnego sprzętu (przeważnie zagranicznego), z drugiej strony tworzy nowy popyt na mikrokomputery.

Tablica 2.2 charakteryzuje mikrokomputery zagraniczne z podziałem na kilka grup w tym przejściowe pomiędzy profesjonalnymi a nieprofesjonalnymi. Wyróżniono ponadto mikrokomputery przenośne jako specyficzną grupę urządzeń, które przy niewielkich rozmiarach i ciężarze, łatwo przenosić, a poprzez modemy łączyć z innymi komputerami lub ich sieciami. Ciekawe właściwości posiadają mikrokomputery nieprofesjonalne, których zasoby dorównują zasobom niezbędnym dla zakwalifikowania ich do grupy profesjonalnych w tablicy 2.1, co powoduje że znajdują one liczne zastosowania w przedsiębiorstwach (np. mikrokomputery Commodore C64 lub Amstrad CPC 464). U podstaw przedstawionego podziału leży, nie znajdujące odbicia w tablicy, oprogramowanie specjalizowane na zastosowanie domowe lub biurowe oraz możliwości dodatkowego wyposażenia zewnętrznego mikrokomputerów.

3 Oprogramowanie systemów mikrokomputerowych

OPROGRAMOWANIE decyduje o możliwościach zastosowania mikrokomputerów. Oprogramowanie podstawowe w postaci systemów operacyjnych zapewnia efektywną eksploatację mikrokomputera, natomiast oprogramowanie narzędziowe umożliwia łatwe budowanie systemów użytkowych.

SYSTEMY operacyjne umożliwiają automatyczne zarządzanie zasobami mikrokomputera w ten sposób, że użytkownik stosunkowo łatwo może wykonywać swoje prace, a jednocześnie zapewniona jest optymalizacja pracy całego systemu.

Podstawowym systemem operacyjnym dla mikrokomputerów 8-bitowych jest system CP/M (Control Program for Microcomputers). Zbudowany jest z trzech modułów: CCP, BIOS i BDOS.

Moduł CCP (Console Command Processor) odpowiada za konwersację z użytkownikiem, interpretuje i wykonuje komendy wprowadzane z klawiatury, ładuje i inicjuje wykonanie programów użytkowych, zleca wykonanie operacji wejścia-wyjścia.

Moduł BIOS (Basic Input/Output System) **stanowi wymienny składnik CP/M uzależniony od konkretnych konfiguracji mikrokomputera a zarządza urządzeniem wejścia-wyjścia.**

Moduł BDOS (Basic Disk Operating System) jest odpowiedzialny za zarządzanie **pamięcią na dyskach elastycznych.**

Znanych jest wiele edycji systemu CP/M, jak również systemy oparte na jego filozofii, a umożliwiających zaspokojenie różnorodnych potrzeb użytkowników (CP/M-86 dla procesorów 16-bitowych, Concurrent CP/M pozwala na współbieżność kilku programów, MP/M umożliwia jednocześnie pracę wielu użytkowników, CP/NET umożliwia łączenie mikrokomputerów w sieci).

Podobną rolę jak system CP/M dla mikrokomputerów 8-bitowych, dla mikrokomputerów 16-bitowych spełnia system operacyjny MS-DOS (Microsoft-Disk Operating System), mający dużo cech wspólnych z CP/M.

System operacyjny UNIX stanowi przykład przystosowania systemu opracowanego dla komputerów do możliwości systemów mikrokomputerowych. System UNIX staje się standardem dla mikrokomputerów 16- i 32-bitowych. UNIX składa się z dwóch warstw: zewnętrznej i wewnętrznej. Warstwę zewnętrzną tworzą programy systemowe (edytory, kompilatory języków, programy sortujące itp.) oraz specjalny program interpreter komend, będący odpowiednikiem modułu CCP w CP/M. Warstwa wewnętrzna zwana jądrem realizuje podstawowe funkcje **systemu operacyjnego: operacje wejścia-wyjścia, obsługa przerwań, gospodarka pamięcią operacyjną i zewnętrzną itp. Obok wielu wersji systemu UNIX znane są systemy będące w rzeczywistości jego pewnymi mutacjami, a mianowicie systemy operacyjne XENIX i TUNIS.**

Oprogramowanie narzędziowe można przedstawić w następujących grupach problemowych:

— **zarządzanie danymi,**

— **przetwarzanie tekstów,**

— **operowanie na arkuszach elektronicznych,**

— **grafika komputerowa,**

a także w postaci zintegrowanych systemów obejmujących wymienione grupy problemowe.

Zarządzanie danymi obejmuje szeroką gamę pakietów programowych i systemów dotyczących przetwarzania kartotek, generatorów sprawozdań, systemów zarządzania bazą danych. Pakiety przetwarzania kartotek umożliwiają tworzenie i aktualizację dokumentów i kartotek, wyszukiwanie w nich informacji, drukowanie, łączenie, sortowanie zbiorów danych (przykład pakietu — FILING ASSISTANT (IBM). Generatory sprawozdań umożliwiają użytkownikowi definiowanie sprawozdań ze zbiorów utworzonych przez inne programy (RAPORTING ASSISTANT (IBM). Najbardziej rozbudowanymi narzędziami zarządzania danymi są relacyjne bazy danych. System zarządzania relacyjną bazą danych umożliwia zakładanie, manipulację i przetwarzanie zbiorów danych w dowolnym obszarze zastosowań — gospodarka materiałowa, fakturowanie, kadry, księgowość. Jako przykłady tych systemów można podać dBASE II, dBASE III, CSK BANK DANYCH.

Przetwarzanie tekstów — są to najogólniej operacje związane z tworzeniem i zapisywaniem informacji tekstowej (pism, dokumentów, artykułów itp.), korektą tekstów, wyszukiwaniem informacji tekstowej, składaniem, szpaltowaniem, redagowaniem tekstów itp. Najbardziej popularnymi pakietami w tym zakresie są WORDSTAR, TEKST CSK, VISIO2 (IBM).

Arkusze elektroniczne wiążą się ściśle z zastosowaniem mikrokomputerów. Oprogramowanie umożliwia tworzenie tablic (arkuszy), a następnie wykonywanie działań arytmetycznych na ich elementach, ustawianie i kasowanie elementów, przemieszczanie wierszy, kolumn, podział tablicy na części, przechowywanie wyników itp. Arkusze elektroniczne znajdują szerokie zastosowanie w kalkulacji kosztów, planowaniu, finansach — do najczęściej stosowanych pakietów zaliczyć można pakiety MULTIPLAN, TABPLAN CSK, VISICALC.

Grafika komputerowa to również jeden z kierunków rozwijających się bardzo szybko zastosowań mikrokomputerów. Umożliwia ona prezentowanie informacji w postaci wykresów różnego typu. Grafika komputerowa korzysta z danych opracowanych przez inne pakiety np. VISICALC, MULTIPLAN, a przykładowymi pakietami grafiki mogą być GRAPHING ASSISTANT (IBM), ZX GRAPH (XEROX).

W systemach zintegrowanych zawarte są omówione już pakiety programowe, które i tak muszą wykorzystywać łączące je powiązanie funkcjonalne i informacyjne. Przykładem takiego systemu może być FRAMEWORK, który obejmuje przetwarzanie tekstów o bogatym zestawie funkcji, kalkulacje tabelaryczne, a także bazę danych będącą odpowiednikiem dBASE III, możliwość okienkowania ekranu monitora i inne jeszcze możliwości. System FRAMEWORK umożliwia integrację sprzętu, oprogramowania danych.

4 Języki programowania

JĘZYK programowania jest to sztuczny język przeznaczony do przedstawiania programów w formie zrozumiałej dla komputera. W praktyce spotyka się bardzo wiele języków programowania, które najogólniej podzielić można na języki wewnętrzne i zewnętrzne. Program napisany w języku wewnętrznym może być wykonany tylko przez komputer określonego typu i nie może być przeniesiony na inny komputer. Programowanie w języku wewnętrznym sprawia dużo trudności, jest pracochłonne i niewygodne. Uproszczenie procesu programowania umożliwiają języki zewnętrzne, wśród których wyróżnić można języki niskiego poziomu i wysokiego poziomu. Przykładem języka niskiego poziomu jest ASSEMBLER. W praktyce za język wysokiego poziomu uważa się język programowania wymagający złożonego translatora, przekształcającego program zapisany w języku źródłowym różnym od języka wewnętrznego maszyny cyfrowej, na postać umożliwiającą jego wykonanie przez komputer. Istnieją dwa zasadnicze typy translatorów: kompilatory oraz interpretatory. Przy technice kompilacyjnej cały program jest najpierw tłumaczony na język wewnętrzny i dopiero wtedy wykonywany, natomiast przy technice interpretacyjnej tłumaczenie programu jest przeplatane z jego wykonaniem — na podstawie wykonania przetłumaczonego fragmentu programu ustalany jest dalszy przebieg procesu interpretacji. Zaletą kompilatorów jest większa szybkość wykonania programu, zaletą interpretatorów jest ich prostsza struktura i możliwość natychmiastowego sprawdzenia zapisanego fragmentu programu. W mikrokomputerach stosuje się zarówno interpretatory jak i kompilatory.

Poniżej przedstawiono krótką charakterystykę najpopularniejszych języków programowania wysokiego poziomu spotykanych w mikrokomputerach. Z uwagi na powszechność stosowania języki BASIC i LOGO zostały scharakteryzowane dokładniej.

PL/M — pierwszy język wysokiego poziomu zastosowany w mikrokomputerach, mikrokomputerowa wersja języka PL/1

FORTH — łączy w sobie elementy języka wysokiego poziomu, języka assemblera, systemu operacyjnego z edytorem.

C — język przeznaczony do programowania systemowego, bardzo podobny do języka maszyny cyfrowej.

PASCAL — język o strukturze blokowej wykorzystywany głównie do nauczania programowania, wywodzi się z języków Algol i PL/1

APL — język przeznaczony do nauczania programowania.

MODULA 2 — zawierą w sobie wszystkie cechy Pascala rozszerzone o różne koncepcje modularności i wieloprogramowości.

COMAL — język programowania zawierający elementy Pascala, Moduli 2, PL/1, LOGO i języka C.

ADA — uniwersalny język programowania posiadający wiele cech Pascala (mniej Algolu i PL/1); najlepiej opracowąny i najbardziej złożony język programowania.

PROLOG — język programowania bardzo wysokiego poziomu, posługuje się terminami niezależnymi od komputera; wykorzystywany w badaniach z zakresu sztucznej inteligencji.

ELEMENTY JĘZYKA SINCLAIR — BASIC

WPROWADZENIE

NAZWA języka BASIC powstała w wyniku złożenia pierwszych liter pełnobrzmiącej nazwy angielskiej „Beginner's „All — purpose Symbolic Instruction Code", co oznacza wielozadaniowy język symbolicznych instrukcji dla początkujących. Język ten został opracowany w 1965 roku przez grupę pracowników College'u w Dortmouth. W pierwszej wersji był to język algorytmiczny przeznaczony do rozwiązywania różnych zadań numerycznych z niedużą ilością danych wejściowych w trybie konwersacyjnym. Przez wiele lat BASIC był podstawowym językiem programowania minikomputerów, dla których opracowywano nowe jego wersje. Szczyt popularności osiągnął wraz z rozwojem techniki mikroprocesorowej i pojawieniem się mikrokomputerów. Praktycznie każdy typ mikrokomputera „ma" własny, niepowtarzalny BASIC, różniący się mniej lub bardziej od innych wersji. SINCLAIR-BASIC przeznaczony jest na mikrokomputery ZX SPECTRUM i ZX SPECTRUM +. Wyróżnia się od innych tym, że wszystkie jego słowa kluczowe traktowane są przez interpreter tak jak cyfry, litery czy znaki specjalne, czyli jako pojedyncze znaki o określonych kodach. Na standardowej klawiaturze „upakowano" powiększony zbiór znaków; stąd tzw. tryby kursowe pozwalające uzyskiwać różne znaki przez naciśnięcie tego samego klawisza.

1. Tryby kursora

WYRÓŻNIAMY pięć trybów kursora. Każdy tryb zawęża zbiór znaków dostępnych z klawiatury. Nazwy typów pochodzą od liter ukazujących się na migającym kursorze:

„K" — tryb słowa kluczowego (ang.: Keyword mode),
„E" — tryb rozszerzony (ang.: Extended mode),
„L" — tryb liter małych (ang.: Letter mode),
„C" — tryb liter dużych (ang.: Capitals mode),
„G" — tryb grafiki (ang.: Graphics mode).

Pracę z rozszerzeniem, sygnalizowaną kursorem „E" uzyskuje się poprzez równoczesne naciśnięcie klawiszy CAPS SHIFT oraz SYMBOL SHIFT. Ponowne wciśnięcie klawiszy SHIFT spowoduje powrót klawiatury do pracy z literami (Kursor „L"). Przełączenia klawiatury na pracę graficzną dokonuje się poprzez jednoczesne naciśnięcie klawisza CAPS SHIFT łącznie z klawiszem 9. Powrót kursora „G" do pracy z literami powoduje jednoczesne wciśnięcie klawiszy 9: CAPS SHIFT Migający kursor "K" pojawia się zawsze na początku linii oznaczający „słowo kluczowe". Gdy już wprowadzimy słowo kluczowe komputer zmienia kursor na „L" — czyli na pracę z literami. Jeśli użytkownik chce wpisać duże litery, wówczas przytrzymuje klawisz CAPS SHIFT i wpisuje żądane litery. Jeśli chce wprowadzić większą liczbę dużych liter bez przetrzymywania klawisza CAPS SHIFT, uprzednio naciska klawisze CAPS SHIFT i 2.

Migający kursor „L" zostaje zastąpiony kursorem „C". Powyższe zasady odnoszą się do ZX Spectrum. Wersja PLUS posiada osobne klawisze, przez jednokrotne naciśnięcie których uzyskujemy zmiany trybów pracy komputera. Są to klawisze: EXTEND MODE dla trybu „E", CAPS LOCK dla trybu „C" i GRAPH dla trybu „G".

Poza wymienionymi klawiszami funkcyjnymi trybu kursora, do wybierania znaków znajdujących się na klawiaturze, służą przełączeniowe klawisze funkcyjne: SYMBOL SHIFT i CAPS SHIFT. Sposób wybierania znaków z klawiszy literowych (od góry: 2, 3 i 4 rząd) różni się od sposobu wybierania znaków z klawiszy numerycznych (od góry: 1 rząd). Różnice te przedstawione są na przykładzie klawisza „Q" i klawisza „3".

Z klawisza „Q" możemy uzyskać 7 różnych znaków, są to znaki:



1. PLOT (słowo kluczowe) — kursor [K]
2. SIN (słowo kluczowe) — kursor [E]
3. ASN (słowo kluczowe) — kursor [E]
4. q Kursor [L]
5. Q Kursor [L] CAPS SHIFT lub kursor [C]
6. <= Kursor [K], [L] lub [C] (SYMBOL SHIFT)
7. znak graficzny zdefiniowany przez programistę lub znak Q — Kursor [G]

Z klawisza „3" możemy uzyskać 5 różnych znaków, a ponadto, za pomocą kodu sterującego kolor karmazynowy tła lub tuszu pozycji znakowych. Są to znaki:



1. 3 — Kursor [K] lub [L] lub [C]
2. # — Kursor [K] lub [L] lub [C] (SYMBOL SHIFT)
3. Kolor tła — Kursor [E] Kolor tuszu — Kursor [E] z CAPS SHIFT
4. LINE — (słowo kluczowe) — Kursor [E] (SYMBOL SHIFT)
5. ▀ — Kursor [G] lub ▄ — Kursor [G] z CAPS SHIFT

Podczas edycji (pisania i poprawiania) linii programu następuje automatyczne przejście kursora z trybu „K" na tryb „L" lub „C" i na odwrót.

2. Organizacja ekranu. Znaki graficzne definiowane przez programistę

EKRAN monitora lub telewizora połączonego z mikrokomputerem ZX Spectrum składa się z trzech części:

- obrzeża ekranu (ang.: BORDER),
- części użytkowej, będącej tłem (ang.: PAPER) dla wyświetlanych znaków,
- części systemowej, do której — zasadniczo — programista nie ma dostępu.

Ekran może być jednocześnie obsługiwany w dwóch trybach:

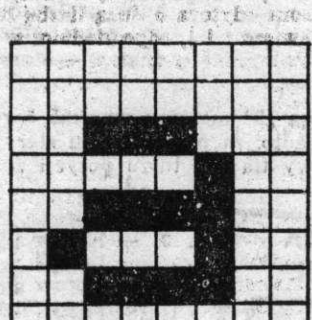
1) jako siatka graficzna o małej rozdzielczości,
2) jako siatka graficzna o dużej rozdzielczości.

Siatka graficzna o małej rozdzielczości dzieli część użytkową i systemową na 24 linie i 32 kolumny. Licząc od góry, 22 pierwsze linie tworzą część użytkową o numerach od Ø do 21. Dwie ostatnie linie tworzą część systemową i oznaczone są #Ø i #1. We wszystkich liniach kolumny numerowane są od Ø do 31.

Najmniejszą jednostką graficzną siatki o małej rozdzielczości jest pozycja znakowa, jako przecięcie jednej linii i jednej kolumny. Wielkość pozycji znakowej jest równa wielkości kursora. W lewym górnym rogu części użytkowej ekranu znajduje się pozycja znakowa (0, 0), w prawym dolnym rogu tej części — pozycja (21, 31).

Dokładne przyjrzenie się (najlepiej przez lupę) jednej pozycji znakowej pozwoli stwierdzić, że składa się ona z 64 punktów (mówimy: pikseli) ułożonych w kwadrat 8×8. Jeden piksel jest najmniejszą jednostką siatki graficznej o dużej rozdzielczości. Obsługa ekranu, w tym trybie ogranicza się tylko do części użytkowej. Nie trudno policzyć, że siatka graficzna o dużej rozdzielczości dzieli część użytkową ekranu na 176 linii i 256 kolumn. Linie numerowane są od Ø do 175, kolumny od Ø do 255. Piksel (Ø, Ø), jako przecięcie zerowej kolumny i zerowej linii znajduje się w lewym dolnym rogu części użytkowej, zaś piksel (255, 175), jako przecięcie linii pionowej i poziomej o podanych numerach zajmuje prawy górny róg tej części.

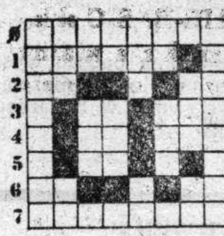
Wiemy już, że każdy pojedynczy znak wyświetlany jest na ekranie w postaci matrycy 8×8 pikseli, zajmującej jedną pozycję znakową. Przykładowo, znak „a" w bardzo dużym powiększeniu wygląda tak:



Literę „a" widzimy na ekranie tylko dlatego, że niektóre piksele są w kolorze tuszu (ang.: INK) a pozostałe w kolorze tła (ang.: PAPER). Gdyby kolory te były jednakowe, to na ekranie nie widzielibyśmy kształtu znaku, chociaż komputer znak ten rozpoznawałby.

Język SINCLAIR-BASIC daje programiście możliwość definiowania własnych znaków graficznych, dostępnych następnie w trybie kursora „G". Programista może jednocześnie (w jednym programie) zdefiniować 21 takich znaków. Liczba ta bierze się stąd, że można definiować („oprogramowywać") graficznie klawisze literowe od „A" do „U". Litery te są właśnie pierwszymi 21 literami w alfabecie łacińskim.

Aby przedstawić metodę definiowania znaków graficznych musimy znacznie wyprzedzić tok nauki. Zdefiniujmy znak alfa („α") na klawiszu „A".



(1)

Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø
1	Ø	Ø	Ø	Ø	Ø	Ø	1	Ø
2	Ø	Ø	1	1	Ø	1	Ø	Ø
3	Ø	1	Ø	Ø	1	Ø	Ø	Ø
4	Ø	1	Ø	Ø	1	Ø	Ø	Ø
5	Ø	1	Ø	Ø	1	Ø	1	Ø
6	Ø	Ø	1	1	Ø	1	Ø	Ø
7	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø

(2)

```
BIN Ø Ø Ø Ø Ø Ø Ø Ø
BIN Ø Ø Ø Ø Ø Ø 1 Ø
BIN Ø Ø 1 1 Ø 1 Ø Ø
BIN Ø 1 Ø Ø 1 Ø Ø Ø
BIN Ø 1 Ø Ø 1 Ø Ø Ø
BIN Ø 1 Ø Ø 1 Ø 1 Ø
BIN Ø Ø 1 1 Ø 1 Ø Ø
BIN Ø Ø Ø Ø Ø Ø Ø Ø
```

(3)

Po wstępnym zakreśleniu kształtu znaku α na matrycy (1) kolorujemy wybrane pola (odpowiadające pikselom). Linie pikseli numerowane są od Ø do 7. Następnie na drugiej matrycy o takiej samej liczbie pól zapisujemy jedynki w pola pokolorowane oraz zera w pola niepokolorowane. Łańcuchy zer i jedynek w liniach tworzą liczby binarne z przedziału (00000000, 11111111). Korzystając z wykazu liczb binarnych (3) wystarczy napisać taki program:

```
1000 POKE USR „a"+0, BIN 00000000
1010 POKE USR „a"+1, BIN 00000010
1020 POKE USR „a"+2, BIN 00110100
1030 POKE USR „a"+3, BIN 01001000
1040 POKE USR „a"+4, BIN 01001000
1050 POKE USR „a"+5, BIN 01001010
1060 POKE USR „a"+6, BIN 00110100
1070 POKE USR „a"+7, BIN 00000000
```

Można zamiast „a" zamiennie używać dużej litery „A".

Napisanie każdej linii musi być zakończone naciśnięciem klawisza ENTER, co jest równoznaczne oznaczeniu końca linii i rozkazowi przesłania do pamięci komputera. Po napisaniu i przesłaniu ostatniej linii należy nacisnąć klawisz „R" (na kursorze musi być litera "K"). U dołu ekranu, w części systemowej, pojawi się słowo kluczowe RUN, które jest rozkazem (tutaj: komendą) uruchomienia programu. Po naciśnięciu klawisza ENTER, który egzekwuje rozkaz, nie minie nawet kilka sekund, a na miejscu, gdzie poprzednio było słowo RUN, pojawi się komunikat pomyślnego wykonania programu

Ø OK.

Wystarczy teraz nacisnąć klawisz GRAPH, czyli włączyć tryb graficzny oraz nacisnąć klawisz „A". Kursor przesunie się w prawo „pozostawiając" znak „α". Taki efekt widać na ekranie. Jaki jest efekt napisania i uruchomienia tego programu z punktu widzenia pamięci komputera? Otóż najwyższą część pamięci RAM zajmuje pamięć grafiki definiowanej przez użytkownika. Słowo POKE tłumaczymy: „wetknij, wepchnij, wciśnij", słowo USR (skrót od USER) tłumaczymy „użytkownik". Osiem cyfr binarnych (bitów: 0 lub 1) tworzy jeden bajt. Nasz program zawiera osiem rozkazów „wciskania bajtu określonego przez użytkownika" do adresów pamięci

„graficznej" komputera przeznaczonych na klawisz „a". Treść programu jest ciągle pamiętana przez komputer i nawet usunięcie programu z pamięci (rozkazem NEW) nie kasuje znaku graficznego „a". Można go, przykładowo, stosować w opisach i komentarzach programów.

3. Struktura elementarna języka

ALFABET języka SINCLAIR-BASIC składa się z 244 znaków. Każdy znak ma własny niepowtarzalny kod zawarty w przedziale ⟨32, 255⟩. Kod 32 odpowiada spacji, kod 255 zaś słowu kluczowemu COPY. Wszystkie znaki można wyświetlić na ekranie za pomocą uruchomienia następującego programu: 1Ø. FOR i = 32 TO 255: PRINT i, CHR$ i : NEX i

PODSTAWOWYMI znakami alfabetu języka są słowa kluczowe. Jest ich dokładnie 88. W zależności od kontekstu użycia, każde słowo kluczowe można przypisać do jednego lub więcej z następujących pięciu rodzajów słów:

1. **Komendy,**
2. **Instrukcje,**
3. **Funkcje standardowe,**
4. **Operatory logiczne,**
5. **Pomocnicze słowa kluczowe.**

Komendy i instrukcje nazywamy łącznie rozkazami, gdyż tylko wystąpienie tych słów powoduje wystąpienie jakiegoś działania komputera.

Komendą nazywamy rozkaz użyty w trybie bezpośrednim. Rozkaz ten jest wykonywany niezwłocznie po przesłaniu go do pamięci (klawisz ENTER). Przykłady komend: RUN, LOAD, COPY, SAVE, PRINT, LET, BORDER.

Instrukcją nazywamy rozkaz użyty w trybie programowym, czyli występujący w linii programu. Rozkaz ten jest wykonywany tylko po uruchomieniu programu komendą RUN lub GO TO.

Funkcją standardową nazywamy słowo kluczowe tworzące wartość określonego rodzaju. Jest zawsze częścią rozkazu. Przykłady: INT, RND, CHR$, PEEK, SIN.

Operatorem logicznym nazywamy słowo kluczowe używane do budowania wyrażeń logicznych w rozkazach. Operatorzy logiczni służą do określania lub zmiany prawdy logicznej warunków. Język SINCLAIR-BASIC zawiera trzy operatory logiczne: AND, OR i NOT.

Pomocnicze słowa kluczowe stanowią syntaktyczne uzupełnienie rozkazów. Przykłady: TAB, AT, LINE, STEP.

Zmienną nazywamy połączenie (logiczne) miejsca w pamięci komputera z wartością. Do wartości zmiennej odwołujemy się za pomocą nazwy zmiennej. Nazwy zmiennych muszą być tworzone według ściśle określonych reguł. Zbiór znaków, które wolno użyć w nazywaniu zmiennych obejmuje:

— **litery duże lub małe, pochodzące z alfabetu łacińskiego,**
— **cyfry,**
— **znak dolara ($).**

Nazwy zmiennych liczbowych budujemy z liter i cyfr. Długość jest w zasadzie dowolna. Pierwszym znakiem nazwy musi być litera. Duże i małe litery nie są rozróżniane; duże litery zamieniane są w pamięci komputera na litery małe.

Nazwy zmiennych tekstowych można budować tylko z jednej litery oraz znaku dolara. Jednocześnie można zatem przetwarzać 26 różnych zmiennych tekstowych. Ograniczenia o podobnym charakterze dotyczą nazywania zmiennych indeksowanych.

Maksymalna długość tekstów wynosi 255. ZX SPECTRUM + pamięta liczby z dokładnością do 9 lub 10 cyfr. Przetwarzać można następujące liczby:

— największe co do wartości bezwzględnej: około 10^{38}
— najmniejsze co do wartości bezwzględnej: około 4×10^{-39}

4. Struktura elementarna programu. Edytor

PROGRAM języka SINCLAIR-BASIC składa się z linii numerowanych liczbami naturalnymi z przedziału ⟨1, 9999⟩. W jednej linii programu można zapisać minimalnie 2 znaki, maksymalnie 255 znaków. Z uwagi na fakt, iż pojedynczym znakiem jest również każde słowo kluczowe, ograniczenie o długości linii jest praktycznie nieistotne. We wspomnianych granicach linia programu może składać się z dowolnej liczby instrukcji, oddzielanych dwukropkiem. Ponadto instrukcje PRINT, LPRINT, INPUT, DRAW, PLOT i CIRCLE mogą zawierać podinstrukcje (tzw. instrukcje lokalne), które oddzielamy średnikiem. Rozkazem przesłania linii do pamięci komputera (po jej napisaniu) jest przyciśnięcie klawisza ENTER. Przesłanie linii do pamięci jest równoznaczne ze skasowaniem przesłanej linii o tym numerze. Ostatnia przesłana linia programu jest wyróżniona tzw. kursorem edytora. Jest to znak większości („>") umieszczony bezpośrednio po numerze linii. Kursor edytora wskazuje, którą linię aktualnie możemy wyedytować, czyli zdublować w części systemowej ekranu celem poprawienia jej. Edycję linii umożliwiają lub ułatwiają następujące klawisze funkcyjne: EDIT, DELETE, ←, →, ↑ i ↓. Klawisz EDIT powoduje zdublowanie linii wyróżnionej w części systemowej ekranu. Linia ta pozostałe ciągle w niezmienionej postaci w listingu programu. Jednokrotne przyciśnięcie klawisza DELETE pozwala skasować jeden znak znajdujący się bezpośrednio na lewo od kursora systemowego. Klawisze ← i → umożliwiają przesuwanie tego kursora — odpowiednio — w lewo i w prawo, w ramach edytowanej linii. Do przesunięć kursora edytora o dużą liczbę linii, programu służą klawisze ↑ i ↓, odpowiednio: w górę, do linii o bezpośrednio mniejszym numerze i w dół, do linii o bezpośrednio większym numerze. Jeśli — w przypadku długich programów — konieczne jest przesunięcie kursora edytora o dużą liczbę linii, to używanie klawiszy ↑ i ↓ może być czasochłonne i kłopotliwe. Wówczas najprościej jest zapisać słowo kluczowe (rozkaz) LIST i numer żądanej linii oraz nacisnąć klawisz ENTER. Kursor edytora automatycznie ustawi się na wskazanej linii. Jeżeli na dole ekranu nie pojawiło się pytanie „scroll?" (pytanie o przesunięcie zawartości ekraru), to wystarczy przycisnąć klawisz EDIT. W przeciwnym przypadku należy uprzednio przesłać odpowiedź odmowną jednym z następujących sposobów: przycisnąć klawisz „N", spacji lub BREAK, lub przesłanie słowa kluczowego STOP.

Po wprowadzeniu do linii poprawek przyciskamy klawisz ENTER. Oznacza to zakończenie edycji, skasowanie poprzedniej postaci linii i przesłanie do pamięci jej nowej postaci.

Najprostszą metodą kasowania całych linii programu jest zapisanie numeru linii i naciśnięcie klawisza ENTER.

5. Obsługa ekranu. Wprowadzenie wyników na drukarkę

ROZKAZEM pisania na ekranie jest instrukcja /komenda PRINT. Słowo kluczowe PRINT może samo stanowić rozkaz, oznacza wtedy tzw. pusty PRINT. Częściej jednak po PRINT występuje przynajmniej jeden element wyświetlany. Mogą to być:

— stałe tekstowe (np. PRINT „Wynik, y = "),
— stałe liczbowe (np. PRINT 12),
— zmienne (np. PRINT X),
— wyrażenia (np. PRINT-b-SQR delta/ (2*a).

Miejsce ekranu (czyli pozycje znakowe), w którym ma być wyświetlany dany element można określić za pomocą znaków interpunkcyjnych: przecinek (,), średnik (;) i apostrof (') i za pomocą słów kluczowych: TAB (tabulacja)) i AT (dokładne wskazanie miejsca ekranu). Przecinek powoduje wyświetlenie od początku następnej wolnej połowy ekranu (czyli od kolumny 0 lub 16). Średnik spełnia dwie funkcje: (1) powoduje wyświetlanie bez odstępu, (2) oddziela wzajemnie podinstrukcje i pomocnicze słowa kluczowe oraz oddziela je od wyświetlanych elementów. Apostrof oznacza przeniesienie wyświetlania do początku następnej linii ekranu; „n" apostrofów zapisanych kolejno daje zatem „n−1" wolnych linii. Pomocnicze słowo kluczowe TAB określa nr kolumny w bieżącej lub następnej linii ekranu (np. PRINT TAB 10; „ZX"). Pomocnicze słowo kluczowe AT określa dokładnie numer linii i numer kolumny. Po słowie kluczowym AT muszą wystąpić dwie wartości liczbowe oddzielone przecinkami. Pierwsza musi zawierać się w przedziale ⟨0,21⟩, jako nr linii. Druga liczba musi zawierać się w przedziale ⟨0, 31⟩, jako nr kolumny.

W rozkazie PRINT mogą wystąpić następujące podinstrukcje określające atrybuty wyświetleń (w nawiasach wartości jakie mogą wystąpić po tych słowach):

— kolor tuszu — INK (0—9),
— kolor tła — PAPER (0—9),
— atrybut migotania — FLASH (0, 1 lub 8),
— atrybut jaskrawości — BRIGHT (0, 1 lub 8),
— atrybut zmiany kolorów tuszu i tła — INVERSE (0 lub 1),
— opcja kasowania poprzednich znaków w przypadku używania słowa AT — OVER (0 lub 1).

Każde z wymienionych wyżej słów można używać jako samodzielne rozkazy i mówimy wtedy o oddziaływaniu globalnym tych rozkazów. Użyte np. w taki sposób:

30 PRINT AT 10, 15; FLASH 1; PAPER 2; INK 7; „ZX"; FLASH 0; BRIGHT 1; INVERSE 1; „SPECTRUM"

mogą oddziaływać lokalnie i dotyczyć tylko elementów wyświetlanych w danym rozkazie PRINT.

Po słowach INK i PAPER mogą wystąpić następujące wartości oznaczające, odpowiednio, dla tuszu lub tła:

0 — kolor czarny,
1 — kolor niebieski,
2 — kolor czerwony,
3 — kolor purpurowy (karmazynowy),
4 — kolor zielony,
5 — kolor cyjankowy (niebiesko-zielony),
6 — kolor żółty,
7 — kolor biały.

Wartości 0 i 1 występujące po słowach FLASH, BRIGHT i INVERSE oznaczają zawsze odpowiednio — wyłączenie i włączenie danego efektu. Jedynym rozkazem dotyczącym kolorów, który nie może być użyty lokalnie jest rozkaz BORDER. Po słowie tym mogą wystąpić wartości kodów kolorów ⟨0 ÷ 7⟩ i dają w efekcie zadany kolor obrzeża ekranu.

Rozkazem podobnym do PRINT jest rozkaz LPRINT. Powoduje wydruk na drukarce. Po LPRINT mogą wystąpić wszystkie wymienione znaki interpunkcyjne oraz słowa TAB, INVERSE i OVER. Nie jest błędem użycie słowa AT, lecz numer linii jest ignorowany, liczy się tylko numer kolumny. Kopię części użytkowej ekranu na papierze drukarki można wykonać za pomocą komendy COPY. Komenda COPY wymaga wstępnego zatrzymania programu: klawisz BREAK, rozkaz STOP lub „naturalne" zatrzymanie programu.

6. Grafika standardowa. Dźwięki.

W ZBIORZE znaków zapisanych na klawiaturze komputera znajduje się 16 standardowych znaków graficznych — na klawiszach „1"÷„8". Kolejne 8 znaków — poza ośmioma opisanymi — otrzymujemy w trybie graficznym kursora, przytrzymując klawisz CAPS SHIFT poprzez inwersję każdego znaku. Każdy standardowy znak graficzny zajmuje swoją wielkością jedną pozycję znakową ekranu.

Na siatce graficznej o wysokiej rozdzielczości „działają" rozkazy PLOT, DRAW i CIRCLE.

Rozkaz PLOT pozwala zabarwić kolorem tuszu (INK) wskazany piksel ekranu (np. PLOT 128, 88). Obie wartości liczbowe muszą zawierać się w przedziałach — odpowiednio — ⟨0, 255⟩ i ⟨0, 175⟩.

Rozkaz DRAW oznacza kreślenie na ekranie linii prostej lub linii będącej wycinkiem okręgu. W pierwszym przypadku wystarczy zapisać dwie wartości liczbowe

100 DRAW dx, dy

Wartość dx oznacza przyrost wartości współrzędnej poziomej, wartość dy — przyrost wartości współrzędnej pionowej. Tak otrzymany punkt jest punktem, do którego ma być „dociągnięta" linia. Początek linii określa poprzedni rozkaz PLOT, DRAW lub CIRCLE, a jeśli takiego nie było, to jest to punkt o współrzędnych (0, 0).

Jeśli po DRAW występują trzy wartości (oddzielone przecinkami), to ostatnia określa łuk, jaki ma zakreślić „ołówek" dochodząc do punktu końcowego. Wartość ta jest podana w radianach. Linia ta kreślona jest w lewo jeśli wartość łuku jest dodatnia oraz w prawo, jeśli ujemna. Wartość zero daje linię prostą.

Rozkaz CIRCLE umożliwia narysowanie okręgu o zadanym środku i promieniu (np. CIRCLE 128, 88, 87).

Po rozkazach PLOT, DRAW i CIRCLE mogą wystąpić podinstrukcje określające atrybuty wyświetleń. Należy jednak pamiętać, że podinstrukcje te oddziałują na całą pozycję znakową (czyli na 64 piksele) zawierającą kreślony piksel. Ciekawą własność działania OVER (wyświetlanie punktów w kolorze tła, a więc są one niewidoczne) jako „gumki do zmazywania linii" można prześledzić na takim prostym przykładzie:

20 DRAW 50, 50; DRAW OVER 1; 50, 50

ZX SPECTRUM wcale nie musi być niemową, może wydawać dźwięki. Zapewnia to rozkaz BEEP. Może być używany jako komenda i jako instrukcja. Po słowie BEEP muszą wystąpić dwie wartości liczbowe oddzielone przecinkami. Pierwsza musi zawierać się w przedziale ⟨0,10⟩ i określa długość dźwięku w sekundach. Druga musi zawierać się w przedziale ⟨−60,69⟩ i oznacza wysokość względną tonu. Podawana jest w półtonach odległości od środkowego C. Przykładowo: BEEP .5,1 daje ton C # powyżej C trwający przez pół sekundy.

7. Rozkaz przypisania wartości zmiennej (LET)

INSTRUKCJA przypisania wartości zmiennej LET powoduje obliczenie wartości i przypisanie jej zmiennej. Składnia rozkazu LET jest następująca:

LET nazwa zmiennej = wyrażenie

Znak „=" nie oznacza „równa się", nie oznacza równości logicznej. Oznacza operację przypisania (podstawienia). Rozkaz LET czytamy następująco: „oblicz wartość wyrażenia stojącego po prawej stronie znaku „=" i podstaw tę wartość pod zmienną, której nazwę zapisano przed znakiem „=". Z punktu widzenia matematyki nonsensem jest taki zapis:

LET X = X+1

gdyż nie ma takiej liczby rzeczywistej x, która spełniałaby powyższą relację. Powtarzamy: znak „=" w rozkazie LET nie oznacza równości logicznej, oznacza podstawienie. Powyższy rozkaz czytamy następująco: Pobierz wartość zmiennej x, dodaj 1 i wynik umieść w zmiennej x. Symbol x oznacza: po prawej stronie znaku = wartość zmiennej, po lewej stronie — nazwę zmiennej czyli miejsce w pamięci. Rodzaj wyrażenia musi być zgodny z rodzajem zmiennej. Jeśli wyrażenie tekstowe to koniecznie zmienna tekstowa; jeśli zmienna liczbowa, to wyrażenie liczbowe lub logiczne.

8. Rozkaz wczytania danych z klawiatury (INPUT)

Wczytanie danych do pamięci komputera podczas przebiegu programu umożliwia rozkaz INPUT. Składnia rozkazu INPUT jest taka sama jak PRINT. Po słowie INPUT powinna wystąpić jedna (lub więcej) nazw zmiennych.

Wczytanie wartości liczbowych różni się od wczytania wartości tekstowych. W pierwszym przypadku kursor „pytający" o wartość nie jest otoczony znakami cudzysłowia. Wczytać można również dowolne wyrażenie dające liczbę i zawierające zmienne już zainicjowane. Działanie INPUT dla danych liczbowych można przerwać wczytując słowo kluczowe STOP i przyciskając klawisz ENTER. W przypadku wczytywania wartości tekstowych kursor pojawia się pomiędzy znakami cudzysłowia. Wczytać można dowolne znaki z klawiatury, nawet tzw. tekst pusty — przez przyciśnięcie klawisza ENTER. Działanie INPUT dla danych tekstowych można przerwać przez skasowanie znaków cudzysłowia i przesłanie słowa kluczowego STOP. Wszystkie znaki interpunkcyjne (przecinek, średnik, apostrof) mają takie samo działanie jak w PRINT.

Wyrażenia, których wartości muszą być obliczone w trakcie INPUT, a także odwołanie do wartości zmiennej, wymaga użycia nawiasów. Przykładowo:

```
60 INPUT x' S$; „ilość:"; il
70 INPUT X$, (x); (S$+„abc:");a
```

Jednym rozkazem INPUT można wczytać dane ale dla dowolnej liczby zmiennych, przy czym ta sama zmienna może dowolną ilość razy „pobierać" wartości za pomocą INPUT.

9. Sterowanie przebiegiem programu (rozkazy IF... THEN oraz GO TO)

ROZKAZY warunkowe budujemy przy pomocy słów kluczowych IF i THEN. IF zawsze rozpoczyna rozkaz warunkowy i poprzedza warunek logiczny. Słowo THEN mówi jakie czynności mają być wykonane, jeśli dany warunek jest prawdziwy. Przykładowo linię:

```
10 IF a<10 THEN LET a=a+1:PRINT a
```

czytamy: „jeśli wartość zmiennej a jest mniejsza od 10, to powiększ jej wartość o 1 i wydrukuj ją". Po słowie THEN może wystąpić wiele instrukcji (tworzących blok instrukcji), które mają być wykonane w przypadku stwierdzenia prawdy logicznej wyrażenia występującego po IF. Jeśli wartość tego wyrażenia jest 0 (fałsz), to nie wykonywane są instrukcje po THEN, a wykonywane są instrukcje zapisane w linii następnej. W programie:

```
30 LET a = -100
40 LET a = a+1: IF ABSa > 10 THEN PRINT a:
GO TO 40
50 PRINT „KONIEC": STOP
```

linia nr 40 wykonywana będzie tak długo, jak długo ABSa > 10. Jeżeli ABS <= 9, to wykonana zostanie linia nr 50. ABS — to nazwa funkcji standardowej, która oblicza wartość bezwzględną argumentu.

Słowo kluczowe GO TO (rozkaz skoku bezwarunkowego) odsyła działanie programu do początku linii o numerze użytym po tym słowie. Rozkazem GO TO nie wolno wykonywać skoków do środka pętli FOR/ /NEXT oraz środka podprogramów. Instrukcja GO TO powinna być ostatnią instrukcją w ramach linii, chyba że po GO TO zapiszemy komentarz

```
110 PRINT "d jest ujemne. Brak pierwiastków":
GO TO 200:REM SKOK do pytania o kontynuację obliczeń.
```

Instrukcją komentarza jest REM. Po tym słowie można zapisać dowolne znaki. REM nie powoduje żadnej akcji komputera, służy do opisu programu dla potrzeb programisty.

10. Inicjacja zmiennych indeksowych (instrukcja DIM)

Zmienne indeksowe tworzą struktury danych, zwane tablicami. Utworzenie takiej struktury wymaga użycia instrukcji DIM (ang. wymiar). Przykładowo instrukcja

```
10 DIM a (10,10)
```

deklaruje tablicę dwuwymiarową o nazwie a. Zawiera ona 100 elementów, każdy element ma tę samą nazwę, a wyróżnia się go indeksami opisującymi jego względne położenie w tablicy. Język SINCLAIR-BASIC umożliwia deklarowanie tablic o dowolnej liczbie i zakresach wymiarów. Nazwa tablicy jest zbudowana z jednej litery i nawiasów zawierających wykaz wymiarów oddzielonych przecinkami. Do elementów tablic liczbowych odwołujemy się przez podanie nazwy tablicy i indeksów w każdym wymiarze, przykładowo:

```
40 LET a (5,5) = 7 : LET i = 5 : LET a (1,i) = 0
```

Deklaracja tablicy liczbowej powoduje przypisanie jej elementom wartości zero.

Nieco inaczej deklaruje się tablice tekstowe. Nazwy tych tablic składają się z jednej litery oraz znaku $ (dolara). Przykładowo linie:

```
10 DIM a $ (15)
20 DIM b $ (10,5) : DIM C$ (5,20,10)
```

deklarują trzy tablice tekstowe. Tablica b$ zawiera 10 elementów każdy o ustalonej długości 5 znaków. Elementami tej tablicy są zmienne b$(1), b$(2), ..., b$(10).

Wstępnie wszystkie otrzymują wartość " " (tekst pusty). Jednak zapytanie o długość dowolnego elementu [np. PRINT LEN$ b$ (1)] zawsze da w wyniku liczbę 5.

Tablica c$ jest dwuwymiarowa, zawiera 100 elementów, każdy o długości 10 znaków. Są to zmienne, indeksowane: C$(1,1), C$(1,2),... C$(1,20), C$), C$(2,1),... C$(5,20).

W linii 10 deklarowana jest tablica zerowymiarowa a$, zawiera ona tylko jeden element a$, o ustalonej długości 15 znaków.

11. Pętle FOR/NEXT

W JĘZYKU SINCLAIR-BASIC wielokrotne wykonanie pewnego ciągu czynności umożliwiają rozkazy (instrukcje/ komendy) FOR i NEXT. Blok instrukcji zawarty między rozkazem FOR (początek pętli, a odpowiadającym mu rozkazem NEXT (koniec pętli) będzie wykonywany pewną liczbę razy, zgodnie z następującą konstrukcją pętli:

l_1 FOR l = w_1 TO wl_2 STEP wl_3

.

.

.

l_2 NEXT l

gdzie: l — zmienna skalarna liczbowa, której nazwa zbudowana jest z jednej litery alfabetu łacińskiego. Zmienna ta nazywana jest zmienną sterującą pętlą,

l_1, l_2 — numery linii programu, mogą to być te same linie,

wl_1 — wyrażenia liczbowe, tzw. wartość początkowa, dolny limit wartości zmiennej sterującej pętlą,

wl_2 — wyrażenie liczbowe, tzw. wartość końcowa, górny limit wartości zmiennej sterującej pętlą,

wl_3 — wyrażenie liczbowe, tzw. krok przyrostu wartości zmiennej sterującej pętlą.

Jeśli wl_3 jest równe 1 to pomocnicze słowo kluczowe STEP i wartość kroku (wl_3) można opuścić. W rozkazie NEXT odpowiadającym danemu rozkazowi FOR musi wystąpić taka sama nazwa zmiennej sterującej pętlą. Wartości wyrażeń wl_1, wl_2 i wl_3 nie można zmieniać w trakcie wykonywania wnętrza pętli. W tym samym programie można wielokrotnie otwierać i zamykać pętle sterowane przez tę samą zmienną.

Pętle mogą się zagnieżdżać, czyli zawierać się wzajemnie. Maksymalna głębokość zagnieżdżania się pętli w języku SINCLAIR-BASIC wynosi 26, tyle bowiem liter zawiera alfabet łaciński.

12. Wewnątrzprogramowy bank danych

WEWNĄTRZPROGRAMOWY bank danych tworzymy za pomocą instrukcji DATA (ang.: dane). Po słowie kluczowym DATA występuje lista wyrażeń liczbowych lub wyrażeń tekstowych, oddzielonych przecinkami. Dostęp do listy danych zapewnia rozkaz (instrukcja/komenda) READ (ang.: czytaj). Po słowie kluczowym READ występuje wykaz nazw zmiennych, oddzielonych przecinkami. Ilość, rodzaj (liczbowe lub tekstowe) oraz kolejność wartości zapisanych po DATA muszą być zgodne z krotnością wystąpienia instrukcji READ oraz z rodzajem i kolejnością zmiennych, których nazwy występują po READ. Każda instrukcja DATA może być rozbita na kilka osobnych instrukcji DATA. Rozkazy READ „czytają" dane kolejno z list danych według wzrastającej numeracji linii programu i zawsze od strony lewej do prawej. Słowo RESTORE przestawia wskaźnik czytania wartości na początek listy danych instrukcji DATA, o numerze linii podanej po słowie RESTORE. Jeśli po RESTORE nie zapiszemy żadnej wartości, to wskaźnik danych przedstawiany jest na początek listy pierwszej instrukcji DATA.

13. Podprogramy

PODPROGRAMY organizuje się za pomocą dwóch rozkazów: GOSUB (ang.: GO to SUBroutine = = przejdź do podprogramu) i RETURN (wróć do programu głównego). GOSUB oznacza wywołanie podprogramu. Po tym słowie kluczowym musi wystąpić numer linii rozpoczynającej podprogram. Powrót z podprogramu (RETURN) wykonywany jest do instrukcji następującej po rozkazie GOSUB wywołującym ten podprogram.

14. Funkcje użytkownika (DEF FN/FN)

JĘZYK SINCLAIR-BASIC oferuje programiście wiele skutecznych funkcji standardowych. Z pewnością nie wystarczą one w przypadku działań na funkcjach matematycznych. Jest, przykładowo, funkcja obliczająca logarytm naturalny (LN), lecz mogą być potrzebne funkcje obliczające logarytm o innych podstawach.

Własną funkcję definiuje się za pomocą instrukcji DEF FN. Składnia tej instrukcji jest następująca:

a) DEF FN litera ([litera] [, litera]) = wyrażenie liczbowe,

b) DEF FN litera $ ([litera $] [litera] [, litera] [, litera $]) = wyrażenie tekstowe

gdzie nawiasy kwadratowe oznaczają elementy definicji nieobowiązkowe. Przypadek a) opisuje definicję funkcji liczbowej, przypadek b) — funkcji tekstowej. Elementy zapisane w nawiasach okrągłych są parametrami funkcji. Nie muszą one wystąpić, lecz trzeba zapisać przynajmniej parę nawiasów.

Przykładowo linia:

100 DEF FN C ( ) = INT (a+.5)

definiuje funkcję zaokrąglającą liczby rzeczywiste do najbliższych liczb całkowitych. Wywołanie funkcji użytkownika zapewnia słowo kluczowe FN. Po FN należy podać nazwę funkcji oraz w nawiasach listę parametrów formalnych zgodną ilościowo i rodzajowo ze specyfikacją po DEF FN. Przykładowo, jeśli

a = 5.1, to instrukcja

60 LET b = FN c ( )

podstawi pod zmienną b wartość 5

15. Funkcje standardowe

1. Funkcje standardowe tekstowe

CHR$

Każdy znak tworzący strukturę elementarną języka SINCLAIR-BASIC (w tym również każdy znak graficzny zdefiniowany przez użytkownika) ma swój niepowtarzalny kod. Stosując CHR$ i wyrażenie liczbowe dające wartość z przedziału (0,255) otrzymamy konkretny znak lub kod sterujący formatem lub atrybutem wyświetleń ekranowych. Przykładowo, komenda:

PRINT CHR$ 65

wyświetli na ekranie znak "A".

A oto zestawienie niektórych kodów sterujących:

przecinek (PRINT)	CHR$ 6
kursor w lewo	CHR$ 8
kursor w prawo	CHR$ 9
kursor w dół	CHR$ 10
kursor w górę	CHR$ 11
DELETE	CHR$ 12
ENTER	CHR$ 13
INK	CHR$ 16
PAPER	CHR$ 17
FLASH	CHR$ 18

Rozkazy PRINT "A", "B"
PRINT "A"; CHR$ 6; "B"
PRINT "A"+CHR$ 6+"B"

dają taki sam efekt.

INKEY$

Funkcja bezargumentowa badająca, który klawisz został w danym momencie naciśnięty.

SCREEN$

Po słowie kluczowym SCREEN$ muszą być w nawiasach podane numery wiersza i kolumny. W wyniku otrzymujemy znak (wartość tekstową) znajdujący się na tak wskazanej pozycji znakowej ekranu.

STR$

Funkcja ta zamienia liczbę na tekst. Po STR$ musi wystąpić wartość liczbowa

VAL$

Funkcja ta daje wartość tekstową jako wynik wyrażenia tekstowego.

Przykład:

```
10 LET a$ = "DO":LET b$ = "RO"
20 LET c$ = "a$+b$" : PRINT VAL$C$
```

W wyniku działania drugiej instrukcji w linii otrzymujemy na ekranie tekst: "DORO".

2. Funkcje liczbowe
ABS

Oblicza wartość bezwzględną argumentu, którego jest wartością liczbową. Jeżeli wartość ta zadana jest wyrażeniem, to należy używać nawiasów.
A C S (ARcus cosinus)

Oblicza wartość kąta na podstawie wartości jego kosinusa. Argumentem funkcji jest wartość liczbowa z przedziału (−1,1). Wynik (kąt) jest mierzony w radianach.

Radiany można zamieniać na stopnie mnożąc wynik przez 180/PI.

ASN (Arcus sinus)
ATN (Arcus tangens)
ATTR (Atrybuty pozycji znakowej)

Funkcja ta określa atrybuty zadanej pozycji znakowej ekranu. Atrybutami są statusy: INK, PAPER, BRIGHT I FLASH. Argumenty funkcji muszą być w nawiasach oddzielone przecinkami. Pierwszy oznacza nr linii a drugi — numer kolumny ekranu. W wyniku otrzymujemy liczbę naturalną z przedziału 0,255 obliczoną według schematu:

Kod koloru tuszu 0—7
plus
8* Kod koloru tła 0—7
plus
jaskrawość 0 (wyłączona) lub 64 (włączona)
plus
migotanie 0 (wyłączone) lub 128 (włączone)

BIN
Zamienia liczby binarne na liczby dziesiętne. Argumentem funkcji jest liczba binarna składająca się z maksymalnie szesnastu zer lub jedynek.
CODE (kod znaku)

Jest funkcją odwrotną do CHR$. Argumentem funkcji jest wartość tekstowa, a wynikiem liczba z przedziału (0, 255) oznaczająca kod pierwszego znaku tworzącego argument.
COS (kosinus)

Oblicza kosinus kąta. Argumentem jest wartość liczbowa będąca kątem mierzonym w radianach. W wyniku otrzymujemy wartość liczbową z przedziału (−1,1).
EXP (Exponent)

Funkcja matematyczna podnosząca liczbę Nepera (e = 2.7182818) do potęgi zadanej argumentem.
INT (całkowity)

Funkcja zaokrąglająca w dół. Argumentem funkcji jest liczba rzeczywista. Wynikiem jest wartość liczbowa — największa liczba całkowita nie większa niż zadana liczba rzeczywista.
LEN (długość tekstu)

Funkcja mierząca długość tekstów. Argumentem jest wartość tekstowa. W wyniku otrzymujemy liczbę całkowitą z przedziału (0,255).
LN (logarytm naturalny)

Funkcja oblicza logarytm naturalny (o podstawie e). Działa jako odwrotność funkcji EXP
PI

Bezargumentowa funkcja liczbowa dająca wartość liczby pi = 3.1415927.
RND (liczba losowa)

Funkcja generuje liczbę losową z przedziału (0,1). Jest funkcją bezargumentową.
SGN (znak)

Funkcja bada znak liczby, jej argumentem jest wartość liczbowa −1,0 lub 1.
SIN (sinus)
SQR (pierwiastek kwadratowy)
TAN (tangens)
USR (podprogram użytkownika).

Funkcja używana do wywołania podprogramu napisanego w kodzie maszynowym umieszczonym w pamięci komputera od określonego adresu. Używana jest również do definiowania grafiki użytkownika i „lokowania" jej w oddzielnym obszarze pamięci.

Przykład wywołania podprogramu napisanego w kodzie maszynowym:

```
100 RANDOMIZE USR 65000
```

VAL (wartość liczbowa)
Funkcja zamienia tekst utworzony z wartości liczbowych na liczbę. Po słowie kluczowym VAL musi wystąpić wartość tekstowa. Jeżeli wartość ta jest zadana stałą lub zmienną, to funkcja VAL „ucina" znaki cudzysłowia i tak otrzymana reszta jest obliczana jako wyrażenie liczbowe, dając stałą liczbową.

Przykładowo, komenda:

```
PRINT VAL "12.3"
```

wyświetli 12.3

Funkcja VAL może również obliczać wyrażenia. Przykładowo:

```
200 LET f$ = "x↑2—3" : LET x = 1
210 PRINT VALf$
```

Funkcja VAL „obcina" znaki cudzysłowia, podstawia pod zmienną X jej aktualną wartość i daje stałą liczbową (efekt działania linii 210) — 1.

Elementy Języka LOGO

LOGO jest uniwersalnym językiem programowania szczególnie przydatnym dla początkujących i zalecanym jako język pierwszego kontaktu z komputerem. Bardzo dobrze opracowana struktura języka czyni go łatwym i wygodnym w użyciu.

JAKO podstawowe właściwości języka LOGO wymienione są:

- **koncentracja uwagi programującego na rozwiązywanym problemie bez wnikania w szczegóły techniczne;**
- **natychmiastowe efekty wykonywanych działań;**
- **możliwość wielokrotnego wykorzystania raz zaprogramowanych sekwencji działań;**
- **definiowanie złożonych działań jako elementarnych w postaci procedur;**
- **możliwość tworzenia z istniejących procedur nowych;**
- **tworzenie efektownych rysunków prostymi środkami.**

Właściwości te realizowane są zasadniczo poprzez niezależne definiowanie procedur i tzw. grafikę żółwia.

SZCZEGÓLNĄ uwagę w LOGO zwraca tzw. żółwia grafika, dająca efektowne wyniki już w początkowym okresie nauki języka, zachęcając tym do dalszej pracy. Język LOGO w pełnej wersji ma cechy nieustępujące innym językom programowania. Ważną zaletą LOGO jest możliwość swobodnego operowania na tekstach i strukturach listowych.

Obecnie na świecie istnieje wiele wersji języka LOGO różniących się znacznie w szczegółach. W Polsce najbardziej popularne są wersje na Apple II, Commodore 64 a przede wszystkim na ZX Spectrum. Krótka prezentacja LOGO ograniczająca się do elementów grafiki żółwia odnosi się do wersji opracowanej przez zespół z Polskiego Towarzystwa Informatycznego na mikrokomputer ZX Spectrum. Interpreter tej wersji języka jest globalną przeróbką Sinclair LOGO firmy SOLI-LCSI, a istotną innowacją jest możliwość uzyskiwania polskich liter typu ą, ć, ę itp. Otrzymuje się je przez naciśnięcie klawisza „GRAPH" i pożądanej litery np. dla „ą" klawisz "GRAPH" a potem "a", dla "ć" klawisz "GRAPH" a potem "c" itd.

Po włączeniu mikrokomputera i wczytaniu programu POLSKIE LOGO na ekranie powinien pojawić się napis

© Zbigniew Kasprzycki
POLSKIE LOGO wersja 2.2

poniżej znak zapytania i kwadratowy migający wskaźnik. W prawym dolnym rogu litera l oznaczająca, że z klawiatury wprowadzane będą małe litery.

Napiszmy teraz czyść lub w skrócie cs. Ekran zostanie wyczyszczony, na środku pojawi się żółw (trójkącik skierowany jednym wierzchołkiem do góry i oznaczony poprzeczną kreską). W lewym dolnym rogu będzie znak zapytania, zaś w prawym pozostanie litera l

Żółw może poruszać się o zadaną liczbę kroków i obracać o zadany kąt podawany w stopniach.

Napiszmy naprzód lub w skrócie np oraz liczbę kroków do pokonania i naciśnijmy klawisz ENTER.

np 60 ENTER

Pomiędzy np a 60 musi być odstęp, w przeciwnym wypadku pojawi się komunikat o treści:

Nie wiem jak zrobić NP60

Należy wówczas nacisnąć ENTER, żeby znowu pojawił się znak zapytania z migającym wskaźnikiem i napisać komendę poprawnie.

Żółw przesunął się do przodu o 60 kroków pozostawiając za sobą ślad (rys. 1).



Rys. 1



Rys. 2

Obróćmy teraz żółwia w prawo (prawo lub pw) o 120 stopni,

pw 120 ENTER

po czym przesuńmy o 20 kroków w przód,

np 20 ENTER

obróćmy o 30 stopni w lewo (lewo lub lw)

lw 30 ENTER

i przesuńmy o 30 kroków w tył (wstecz lub ws)

ws 30 ENTER

Wyniki dotychczasowej pracy przedstawia rys. 2.

Chcąc usunąć jakiś fragment linii, trzeba „dać" żółwiowi gumkę za pomocą komendy ścieranie i przesunąć nim po tym fragmencie. Po czym „zabieramy" gumkę komendą opu, która oznacza opuszczenie pisaka. Napiszemy więc (rys. 3)

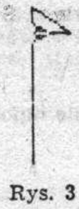
ścieranie np 12 opu ENTER

Jak widać, komendy mogą być pisane w jednym wierszu i oddzielone od siebie odstępami. Naciśnięcie klawisza ENTER powoduje wykonanie całej sekwencji komend.

Żeby wrócić z żółwiem do pozycji wyjściowej w środku ekranu można użyć komendę wróć. Żółw wracając będzie jednak nadal zostawiał za sobą ślad. Dla wyeliminowania tego trzeba najpierw podnieść pisak komendą pod, zaś po komendzie wróć opuścić go przez opu. Napiszmy więc

pod wróć opu ENTER

Po wykonaniu tej sekwencji żółw znów jest w środku ekranu i zwrócony jest do góry (rys. 4).



Rys. 3



Rys. 4

W LOGO istnieje możliwość „nauczenia" żółwia wykonywania danej sekwencji komend, którą możemy wykorzystać później w dowolnym miejscu. Takiej sekwencji komend nadaje się nazwę i określana jest ona mianem procedury.

Zdefiniujmy procedurę o nazwie dom pozwalającą na umieszczenie żółwia w środku ekranu i zwróconego do góry, na wzór ostatniej sekwencji komend.

oto dom ENTER

Teraz pojawia się zamiast znaku zapytania inny znak — znak > oznaczający, że wprowadzone komendy odnoszą się do definiowanej procedury. Wobec tego rysunek na ekranie nie ulegnie zmianie

Napiszmy dalej

pod wróć opu ENTER

a następnie

już ENTER

co spowoduje zakończenie definiowania procedury i pojawienie się komunikatu

dom zdefiniowane.

Możemy teraz pisząc jedynie słowo dom spowodować wykonanie wszystkich komend zawartych w tej procedurze. Procedurę dom wykorzystywać będziemy wielokrotnie, ale nieco później. Dotychczasowym efektem naszej pracy jest maszt z chorągiewką. Narysujemy teraz namiot w postaci trójkąta. W tym celu przesuńmy się w prawo

pw 90 np 30 ENTER

i zdefiniujmy procedurę do rysowania trójkąta, w której wykorzystamy nową komendę — powtórz. Komenda ta pozwala na wykonywanie zadaną ilość razy ciągu komend umieszczonych w nawiasach kwadratowych. W naszym przypadku będzie to trzykrotne przesunięcie się do przodu i obrót o taki kąt, żeby powstał trójkąt.

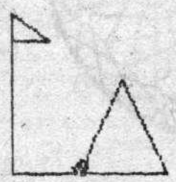
oto trójkąt ENTER

powtórz 3 [np 40 pw 120] ENTER

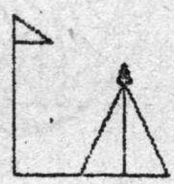
już ENTER

gdzie nawiasy kwadratowe [ ] otrzymujemy naciskając jednocześnie klawisz SYMBOL SHIFT i Y dla [, oraz SYMBOL SHIFT i U dla].

Ustawmy teraz właściwie żółwia i wywołajmy procedurę (rys. 5)



Rys. 5



Rys. 6

lw 60 trójkąt ENTER

Następnie dorysujmy słupek w namiocie (rys. 6)

pw 60 np 20 lw 90 np 35 ENTER

i wróćmy do pozycji początkowej na środku ekranu stosując wprowadzoną już procedurę dom.

dom ENTER

Jak się można było zorientować na końcu wiersza trzeba zawsze nacisnąć klawisz ENTER, dlatego też rezygnujmy z przypominania o tym.

Umieśćmy teraz na rysunku drzewo. Sekwencja komend

pw 90 np 100 lw 90 np 30

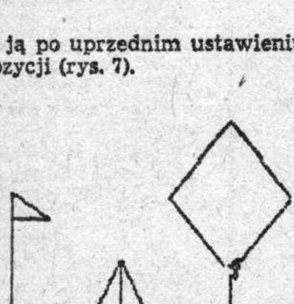
ustawia żółwia w nowej pozycji i rysuje pień drzewa. Koronę drzewa wykonamy w formie kwadratu, definiując go następującą procedurą

oto kwadrat

powtórz 4 [np 40 lw 90]

już

i wywołując ją po uprzednim ustawieniu żółwia we właściwej pozycji (rys. 7).



Rys. 7

pw 45 kwadrat

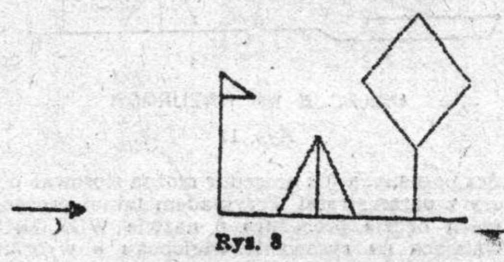
Przejdźmy żółwiem do podstawy drzewa, obróćmy w prawo

lw 45 ws 30 pw 90

i wprowadźmy

np 80

Stało się coś nieoczekiwanego, mianowicie żółw rysując linie wyszedł z lewej strony ekranu. Związane jest to z tym, że ekran jest jakby sklejony brzegami i to zarówno w poziomie jak i w pionie (rys. 8).



Rys. 8

Usuńmy część ostatniej linii

ścieranie ws 60 opu

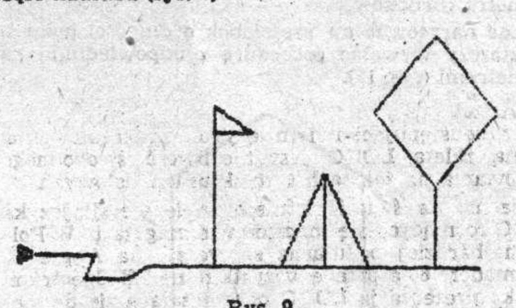
i ustawmy żółwia w nowym miejscu

ws 150

Narysujmy teraz żaglówkę, robiąc przedtem mały uskok

lw 30 ws 10 pw 30 ws 20

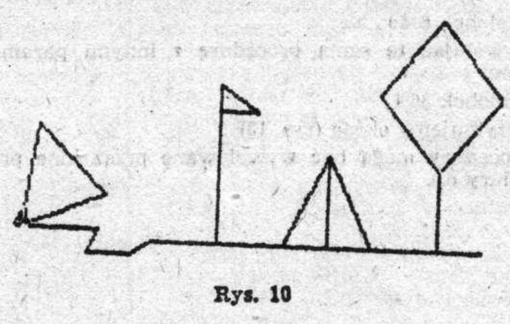
i część kadłuba (rys. 9)



Rys. 9

lw 60 np 10 pw 60 ws 35

Do narysowania żagla wykorzystamy istniejącą już procedurę trójkąta (rys. 10).



Rys. 10

lw 80 trójkąt

Dokończymy rysunek pisząc

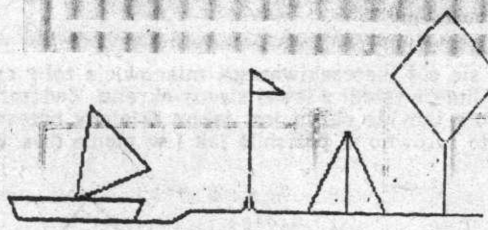
lw 100 np 35 pw 60 ws 10 lw 60 ws 60

i umieścimy żółwia w pozycji wyjściowej (rys. 11)

dom

Możemy zatytułować nasz obrazek. Wykorzystamy do tego komendę kursor [x y] pozwalającą na ustawienie kursora, gdzie x — numer kolumny (0—31) a y — numer wiersza (0—21), oraz komendę pisz (pp) do drukowania tekstu. Wprowadźmy więc

kursor [7 18] pp [WAKACJE NA MAZURACH]
Duże litery uzyskujemy w ten sam sposób jak w BASIC-u, o czym informuje nas litera C w prawym dolnym rogu.



UAKACJE NA MAZURACH

Rys. 11

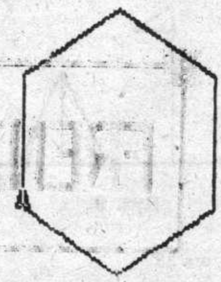
Oprócz poznanych już procedur można stosować procedury z parametrami. Przykładem takiej procedury niech będzie procedura o nazwie WIELOBOK pozwalająca na rysowanie wieloboku o wybranej ilości boków i długości boku.

oto wielobok :ileboków :długość
powtórz :ileboków [np :długość pw 360 / :ileboków]
już

gdzie :ileboków i :długość to parametry, a dwukropek przed ich nazwą oznacza, że chodzi o ich wartość.

Kąt o jaki za każdym razem ma się obrócić żółw wyliczony jest z podzielenia 360 przez wartość parametru :ileboków.

Chcąc narysować np. sześciobok o długości boku 50, wystarczy wywołać procedurę z odpowiednimi parametrami (rys. 12).



Rys. 12

Wielobok 6 50

Wywołując tę samą procedurę z innymi parametrami

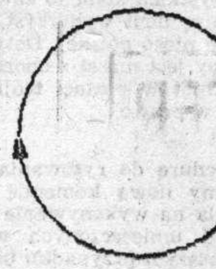
Wielobok 36,4

otrzymujemy okrąg (rys. 13)

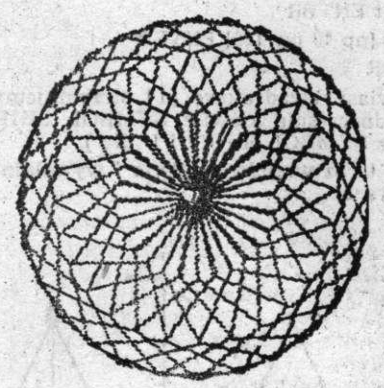
Procedury mogą być wywoływane przez inne procedury np.

oto serweta :ileboków :długość
powtórz 53 [wielobok :ileboków :długość lw 13]
już

Wywołanie tej procedury z parametrami 7 i 30 spowoduje wyrysowanie pięćdziesięciu trzech siedmioboków z przesunięciem co 13 stopni (rys. 14).



Rys. 13



Rys. 14

Możemy wprowadzić procedurę rysującą okrąg o zadanym promieniu. Będzie miała ona postać:

oto okrąg :promień
przypisz "krok :promień * p↑/18
pw 5
36 [np :krok pw 10]
lw 5
już

gdzie komenda przypisz umożliwia nadanie zmiennej krok wartości iloczynu parametru promień przez liczbę pi (ze strzałką w górę). Znak cudzysłowia przed słowem krok oznacza, że chodzi nam o nazwę zmiennej, a nie o jej wartość. Chcąc otrzymać wartość zmiennej krok poprzedzamy ją znakiem dwukropka.

Przedstawione zostały tu jedynie podstawowe elementy języka LOGO, a właściwie "grafiki żółwia". Pełny zakres języka zawiera jeszcze wiele komend realizujących różne funkcje.

5 GRY

MONSTER MUNCHER

GRA polega na zjedzeniu wszystkich jabłek w labiryncie, unikając jednocześnie zjedzenia przez potwory.

Po zjedzeniu specjalnego jabłka-detonatora zyskuje się na pewien czas supermoc, pozwalającą na zjedzenie potwora. Jeśli zjedzone zostaną wszystkie jabłka w labiryncie, gra zaczyna się od nowa w coraz szybszym tempie.

Zaczynając gry wybiera się poziom trudności oraz 1 lub 2 graczy. Liczniki pokazują dotychczasowy najlepszy wynik, wynik aktualny i pozostałą liczbę żyć.

Punktacja:

2000 pkt. premii za ukończenie pierwszej gry i 4000 za każdą następną.

10 pkt. — jabłka
50 pkt. — detonatory
50 pkt. — premia owocowa
100 lub więcej — potwór

GROUND ATTACK

ZADANIEM grającego jest pilotowanie statku kosmicznego poprzez kręte groty oraz niszczenie nieprzyjacielskich pocisków i składów paliwa. Gra ma dziewięć poziomów trudności.

Sterowanie:

5 — przyspieszenie
8 — zwolnienie
6 — lot nurkowy
7 — wznoszenie
0 — strzał z laserów

SPACE RIDERS

GRA polega na obronie Ziemi, za pomocą bazy działek, przed kolejnymi 55 atakami nieprzyjaciół.

Sterowanie:

Z — w lewo
X — w prawo
SPACE — strzał

Punktacja za trafienia:

10 pkt. za dwa niższe rzędy; 20 pkt. za dwa środkowe rzędy; 30 pkt. za wyższe rzędy; 50, 100 lub 200 pkt. za zniszczenie obcego statku kosmicznego.

Za zdobycie 1000 pkt. przyznawane jest dodatkowe działko.

PLANETOIDS

GRAJĄCY musi niszczyć i unikać przelatujących planetoid. Zagrożenie stanowią również nieprzyjacielskie statki kosmiczne rażące seriami bomb.

Sterowanie:

Z — obrót w lewo
X — obrót w prawo
ENTER — siła ciągu
SPACE — strzał
H — nadprzestrzeń

Punktacja za trafienia:

100 pkt. — duża planetoida
200 pkt. — średnia planetoida
300 pkt. — mała planetoida
500 pkt. — latający spodek

Za zdobycie 20 000 pkt. przyznawana jest premia.

MISSILE

ZBLIŻA się pocisk balistyczny. Zadaniem grającego jest zniszczenie go za pomocą podwójnego działka laserowego. Zbliżający się pocisk obserwowany jest przez teleskop, który trzeba nakierowywać na migający punkt.

Sterowanie:

Z — w lewo
X — w prawo
P — w górę
ENTER — w dół
M — powiększenie
N — zmniejszenie
SPACE — strzał

FRENZY

PLANETA robotów ZETUS skazana jest na zagładę, w związku z tym roboty planują zaatakować Ziemię. Zadaniem grającego jest przeniknięcie do kwatery głównej robotów i zniszczenie ich. Mury są pod napięciem i dotknięcie ich grozi śmiercią. Roboty zrobione są z laseroodpornego metalu i jedynie ich głowy są słabym punktem. Przywódca robotów ukazujący się na latającym pojeździe jest niezniszczalny, należy go więc unikać ratując się ucieczką do innej części kwatery głównej.

Sterowanie:

M — w prawo
N — w lewo
S — w górę
X — w dół
A — strzał
H — zatrzymanie gry

JUMPING JACK

ZADANIEM skaczącego Jacka jest skompletowanie wiersza z poszczególnych linijek, ale trzeba mu w tym pomóc. Trzeba przejść przez dwadzieścia ekranów, a w ramach ekranu przez osiem poziomów.

Jacek może przemieszczać się do góry przez poszczególne poziomy, skacząc przez poruszające się dziury. Początkowo występują jedynie dwie dziury. Jacek musi unikać wpadnięcia do dziury uciekając w lewo lub w prawo. Przy czym uciekając na jeden skraj ekranu pokazuje się z drugiego skraju. Wpadając do dziury rozbija się o niższy poziom i pozostaje nieprzytomny przez pewien czas nie mogąc się poruszać. Za każdym razem gdy spadnie na dół ekranu, traci życie. Po każdym udanym skoku pojawiają się nowe dziury — maksymalnie osiem. Po wykonaniu skoku z najwyższego poziomu przechodzi się do nowej części (ekranu). Po pokonaniu pierwszej części, w następnych dochodzą różnego rodzaju niebezpieczeństwa (potworki, lokomotywy itp.), których liczba zwiększa się o jeden z każdym nowym ekranem. Jeśli Jack wpadnie na któreś z nich, to traci przytomność.

Punktacja:

Za skompletowanie jednej zwrotki wiersza przyznawane jest nowe życie. Punkty przyznawane są za każdy udany skok i odpowiednio więcej punktów za każdy nowy ekran.

Sterowanie:

SYMBOL SHIFT — w lewo
SPACE — w prawo
CAPS SHIFT — skok
Z — zatrzymanie gry

PSSST

BOHATEREM gry jest robot Robbie pielęgnujący wzrost kwiatu w swoim ogrodzie. Jego zadanie jest bardzo odpowiedzialne, bowiem na roślinę czyhają groźne szkodniki. Robbie wyposażony jest w trzy puszki opryskiwacza w aerozolu, pojemnik nawozu, siatkę i dmuchawę.

Nasienie umieszczone w ziemi zaczyna wzrastać. Żeby wzrost był szybszy Robbie musi zgromadzić porozrzucane akcesoria (migające) oraz zachować pęd i liście, które pobierają światło słoneczne, wolne od szkodników. Jeśli plantacja ma więcej liści, to szybciej rośnie. Plantacja jest szczególnie atrakcyjna dla szkodników, gdy młody pęd zacznie wydzielać aromatyczny zapach. Szkodniki — Międzygniezdne Ślimaki, Ogniste Gąsiennice i Niebezpieczne Muszki — niszczone są tylko jednym z rodzajów opryskiwacza. W przypadku użycia niewłaściwego opryskiwacza szkodnik zostanie jedynie chwilowo oszołomiony.

Jedynie 20% całej plantacji zakwitnie i urzeczywistni pragnienie Robbiego.

Czas wzrostu wynosi około 2 do 5 minut.

Sterowanie:

Q — w lewo
W — w prawo
E — w górę
R — w dół
T — uruchomienie opryskiwacza
CODS SHIFT — zatrzymanie gry

Na ekranie uwidoczniony jest wynik i ilość pozostałych żyć dla jednego lub dwóch graczy oraz najlepszy dotychczasowy wynik.

COOKIE

CHARLIE — kucharz, przechowuje wszystkie składniki potrzebne do przyrządzania potraw zamknięte w spiżarni. Są one jednak bardzo niesforne i uciekają z szuflad przyciągając za sobą szereg niebezpieczeństw. Zadaniem Charliego jest upieczenie ciasta. Musi więc oszołomić rozbiegane składniki za pomocą bomb z mąki i wrzucić je do miski. Jeśli to mu się nie uda, to wejdą do pojemnika na śmieci i zostaną zjedzone przez potwora.

Należy powrzucać wszystkie składniki do miski, na której umieszczony jest licznik potrzebnych składników, licznik zmienia się w zależności od tego, co zostało w niej umieszczone (składnik lub śmieć), żeby upiec ciasto.

Sterowanie:

Q — w prawo
W — w lewo
E — w dół
R — w górę
T — rzucanie torbami z mąką
CAPS SHIFT — **zatrzymanie gry**

CHEQUERED FLAG

CHEQUERED FLAG jest grą symulującą wyścigi samochodowe. Grający siedzi za kierownicą samochodu wyścigowego Formuły 1. Przed sobą ma kierownicę, deskę rozdzielczą z różnymi przyrządami oraz drogę do pokonania.

Samochód grającego zachowuje się tak jak prawdziwy samochód wyścigowy. Trzeba więc dobierać odpowiednie parametry jazdy, żeby uzyskać jak najlepszy wynik. Elementami wpływającymi na optymalną jazdę są prędkość i obroty. Właściwe operowanie biegami i prędkością zapobiegają przegrzaniu silnika jak i wpadnięciem w poślizg na zakrętach. Można ułatwić sobie zadanie wybierając samochód z automatyczną skrzynią biegów, kosztem zmniejszenia mocy samochodu.

Na desce rozdzielczej umieszczony jest szybkościomierz, obrotomierz, wskaźnik paliwa i temperatury wraz ze wskaźnikami wyboru biegów oraz wyświetlany jest czas okrążenia i ilość okrążeń.

Na torze czyhają na kierowcę różnego rodzaju niebezpieczeństwa przeszkadzające w uzyskaniu rekordowego wyniku.

OLEJ I WODA. Przy przejeżdżaniu przez olej lub wodę nie należy skręcać, bo może to spowodować wpadnięcie w poślizg.

SKRAJ DROGI. Zjechanie z toru powoduje rozbicie się i zakończenie wyścigu.

SZKŁO. Wjechanie na szkło spowoduje przebicie opony.

Trzeba wówczas dowlec się do boksu, gdzie mechanicy zmienią koło.

PRZEGRZANIE. Stałe utrzymywanie za dużych obrotów spowoduje wzrost temperatury, co może być przyczyną wylecenia silnika w powietrze. Można przeciwdziałać temu zjeżdżając do boksu, co szybko ochłodzi silnik. Należy unikać redukowania biegów przy wysokich obrotach silnika, ponieważ to też spowoduje ciężkie przegrzanie.

BOKS NAPRAWCZY. Po poprawnym zjechaniu do boksu dokonywany jest pełny serwis obejmujący wymianę koła, uzupełnianie paliwa i chłodzenie silnika.

Rozpoczynając grę należy wybrać tor, samochód i liczbę okrążeń i dopiero można startować po zapaleniu się zielonego światła.

Sterowanie:

Q — przyspieszenie
I — hamowanie
M i dowolny na prawo od M — zmiana biegów w górę
N i dowolny na lewo od N — zmiana biegów w lewo
A — szybki skręt w lewo
S — wolny skręt w lewo
D — szybki skręt w prawo
F — wolny skręt w prawo
H — zatrzymanie
H i T razem — przerwanie wyścigu

TORY

Do wyboru jest jeden z dziesięciu torów wzorowanych na najsławniejszych torach Grand Prix.

SAMOCHODY

FERETTI TURBO

Bardzo silny samochód z turbodoładowaniem osiąga max. 640 kM mocy przy obrotach 8000—10000 co daje mu ogromne przyspieszenie.

PSION PEGASUS

Nowa konstrukcja osiągająca max. 560 kM przy obrotach 5000—10000 o wyjątkowo niskim współczynniku oporu. Bardzo szybki samochód, a jednocześnie stosunkowo łatwy do prowadzenia.

Mc FASTER SPECIAL

Samochód z automatyczną skrzynią biegów, co czyni go szczególnie przydatnym dla mniej doświadczonych kierowców. Pomimo tego jest szybkim i niezawodnym samochodem osiągającym moc 500 kM.

HUNGARY HORACE

GRAJĄCY jest Głodnym Horacym, który zamierza zjadać wszystko co napotka na swojej drodze Horacy porusza się po ścieżkach, mostach i tunelach parku sterowany następującymi klawiszami:

P — w prawo
I — w lewo
Q — w górę
Z — w dół

Za zjedzenie kwiata przyznawane jest 10 punktów. Trzeba wystrzegać się strażników, którzy chcą złapać Horacego i wyrzucić z parku. Jeśli Horacy zostanie trzy razy złapany, to nie będzie już mógł wrócić do parku i gra się skończy. W parku jest kilka dzwonków alarmowych. Jeśli Horacy uruchomi któryś z nich, to strażnicy wpadną w panikę a Horacy może ich złapać i wyrzucić z parku. Gdy jeden ze strażników upuści swój obiad składający się z wiśni lub truskawek, należy go zjeść za co Horacy otrzyma 100 punktów.

Park podzielony jest na kilka części połączonych między sobą przejściami. Horacy nie musi zjeść wszystkich kwiatów ani wzniecać alarmu przed wejściem do innej części. Każda następna część parku jest trudniejsza od poprzedniej.

HORACE & THE SPIDERS

HORACY wybiera się na polowanie na pająki, wyposażony w cztery fiolki z surowicą chroniącą go przed ukąszeniem przez pająki. Jednak po użyciu surowicy musi rezygnować z polowania.

Zadanie jego jest trzyetapowe: musi wspiąć się na wzniesienie żeby dotrzeć na górę, musi pokonać Pajęczy Most i na końcu musi pozabijać pająki w ich jaskiniach.
Horacy musi przetrwać każdy z tych etapów, żeby przejść do następnego. Po pomyślnym pokonaniu wszystkich etapów i zabiciu pająków powraca do etapu pierwszego zaczynając polowanie od nowa, przy czym dochodzą nowe niebezpieczeństwa do pokonania.

Sterowanie:

Q — poruszanie lub skok do góry
Z — poruszanie lub skok w dół
I — poruszanie w lewo
P — poruszanie w prawo
klawisze V, B, N lub M — powodują tupanie Horacym (jedynie dla 3 etapu)

Wspinanie się na wzniesienie.
Horacy musi wspiąć się na wzniesienie skacząc do góry i następnie dostać się do następnego poziomu unikając jednocześnie pająków. Jedynym sposobem uniknięcia pająków jest przeskakiwanie ich. Naciśnięcie klawisza Q powoduje jedynie skok do góry. Żeby zaś uzyskać skok do przodu trzeba nacisnąć klawisz Q gdy klawisz P jest naciśnięty. Jeśli Horacy źle wyliczy skok, to może stłuc fiolkę z surowicą.

Pokonywanie Pajęczego Mostu
W celu pokonania Pajęczego Mostu, Horacy musi uchwycić się sieci pajęczej (klawisz Q) i następnie przeskakiwać w odpowiednim czasie z nici na nić aż do dotarcia na drugą stronę. **Należy jednak się spieszyć, ponieważ gdy pająki poczują że Horacy wskoczył na nić będą próbowały go wciągnąć. Po osiągnięciu drugiej strony naciśnięcie klawisza Z spowoduje skok w dół.**

Zabijanie pająków
Gdy Horacy dotrze do jaskini, to znajdzie w niej pełno pająków pracowicie tkających olbrzymią pajęczynę. Horacy robiąc w niej dziury, tupiąc po może przejść przez pajęczynę niej (klawisze V, B, N lub M) aż utoruje sobie drogę. Pająki będą próbowały naprawić szkody. Podczas naprawiania pająki są bezbronne i Horacy może je zabić przez tupanie. Jeśli poniżej jest jeszcze jakaś dziura do naprawy, to pająk nie ginie.
Grę można zatrzymać naciskając klawisz S, a następnie wznowić naciskając dowolny klawisz z dolnego rzędu. Przerwanie gry i rozpoczęcie od nowa następuje po naciśnięciu klawiszy G i H.

6 Krótkie programy

DZIEŃ TYGODNIA

CHCĄC wiedzieć jakim dniem tygodnia był dzień ogłoszenia Konstytucji 3 Maja 1791 roku lub jakim będzie 1 stycznia roku 2000, wystarczy wprowadzić i uruchomić poniższy program. Program ten podaje dni tygodnia dla dat z przedziału od 15 października 1582 roku do 31 grudnia 2499 roku.

```
  10 REM ************************
  20 REM *** DZIEN  TYGODNIA ***
  30 REM ************************
  40 DIM A(12): DIM A$(7,12)
  50 FOR I=1 TO 7: READ B$: LET
A$(I)=B$: NEXT I
  60 FOR I=1 TO 12: READ A(I): N
EXT I
  70 PRINT AT 18,7;"PODAJ DATE W
 ZAKRESIE";AT 20,4;"OD 1582.10.1
5 DO 2499.12.31"
  80 INPUT "ROK ";RR: INPUT "MIE
SIAC ";MM: INPUT "DZIEN ";DD
  90 IF RR<1582 OR RR>2499 THEN
RUN
 100 IF RR=1582 AND (MM<10 OR (M
M=10 AND DD<15)) THEN RUN
 110 IF MM<1 OR MM>12 OR DD<1 OR
 DD>A(MM) THEN RUN
 120 IF MM=2 AND DD=29 AND INT (
RR/4)<>RR/4 THEN RUN
 130 LET ND=MM: IF MM<3 THEN LET
 RR=RR-1: LET MM=MM+12
 140 LET ND=INT (365.25*RR)-INT
(RR/100)+INT (RR/400)+31*(MM-1)-
INT (.4*MM+2.3)+DD
 150 LET ND=ND+1721060
 160 LET WT=ND-INT (ND/7)*7+1
 170 CLS : PRINT AT 10,0;"DZIEN
";RR;".";MM;".";DD;" TO ";A$(WT)
 180 PRINT #0;"     NASTEPNA DA
TA ? (T/N)"
 190 LET B$=INKEY$: IF B$="" THE
N GO TO 190
 200 IF B$="T" OR B$="t" THEN CL
S : GO TO 80
 210 DATA "PONIEDZIALEK","WTOREK
","SRODA","CZWARTEK","PIATEK","S
OBOTA","NIEDZIELA"
 220 DATA 31,29,31,30,31,30,31,3
1,30,31,30,31
RUN

      PODAJ DATE W ZAKRESIE

    OD 1582.10.15 DO 2499.12.31

ROK 1875
MIESIAC 5
DZIEN 13

DZIEN 1875.5.13 TO CZWARTEK
     NASTEPNA DATA ? (T/N)
```

LICZBY RZYMSKIE

WSPÓŁCZEŚNIE do zapisywania liczb stosujemy znaki w postaci cyfr arabskich. Dawniej jednak, powszechnie stosowany był zapis za pomocą znaków wprowadzonych przez starożytnych Rzymian i nazwanych od nich cyframi rzymskimi. Przedstawiany program pozwala na zamianę liczby arabskiej z zakresu od 1 do 4999 na liczbę rzymską.

```
  10 REM ************************
  20 REM *** LICZBY RZYMSKIE ***
  30 REM ************************
  40 DIM A(14): DIM R(14): DIM R
$(14,2)
  50 FOR I=1 TO 14: READ A(I): R
EAD A$: LET R$(I)=A$: NEXT I
  60 INPUT "PODAJ LICZBE (1-4999
)"; LINE A$
  70 LET AR=INT VAL A$: IF AR<1
OR AR>4999 THEN GO TO 60
  80 LET B$="": LET A=AR: FOR I=
1 TO 14: LET R(I)=0: NEXT I
  90 FOR I=1 TO 14
 100 LET M=AR-A(I)
 110 IF M>=0 THEN LET R(I)=R(I)+
1: LET AR=M: GO TO 100
 120 NEXT I
 130 FOR I=1 TO 14: FOR J=1 TO 4
 140 IF R(I)>0 THEN LET B$=B$+R$
(I,1): LET R(I)=R(I)-1: IF R$(I,
2)<>" " THEN LET B$=B$+R$(I,2)
 150 NEXT J: NEXT I
 160 PRINT AT 8,3;"LICZBA ARABSK
A = ";A
 170 PRINT AT 10,3;"LICZBA RZYMS
KA = ";B$
 180 PRINT #0;"    NASTEPNA LICZ
BA ? (T/N)"
 190 LET A$=INKEY$: IF A$="" THE
N GO TO 190
 200 IF A$="T" OR A$="t" THEN CL
S : GO TO 60
 210 DATA 1000,"M",900,"CM",500,
"D",400,"CD",100,"C",90,"XC",50,
"L",40,"XL",10,"X"
 220 DATA 9,"IX",5,"V",4,"IV",2,
"II",1,"I"

RUN

PODAJ LICZBE (1-4999)1986

     LICZBA ARABSKA = 1986

     LICZBA RZYMSKA = MCMLXXXVI

       NASTEPNA LICZBA ? (T/N)
```

LUPA

PROGRAM ten powiększa wprowadzony tekst (maksymalnie do 8 znaków) i wyświetla na środku ekranu. Dodatkowo wybrać można kolor tekstu, tła i ramki wprowadzając cyfry od 0 do 7 odpowiadające właściwym kolorom.

```
  10 REM ************************
  20 REM ********* LUPA ********
  30 REM ************************
  40 INPUT "Podaj tekst (max.8 z
nakow)",t$
  50 INPUT "Kolor ramki ? (0-7)
";kr
  60 INPUT "Kolor tla ? (0-7) ";
kt
  70 INPUT "Kolor tekstu ? (0-7)
";kx
  80 BORDER kr: PAPER kt: INK kx
: CLS
  90 PRINT AT 21,0;t$;AT 8,0;
 100 LET p$=""
 110 FOR i=7 TO 1 STEP -2
 120 FOR j=1 TO 63 STEP 2
 130 LET p$=p$+CHR$ (128+POINT (
j,i)+2*POINT (j-1,i)+4*POINT (j,
i-1)+8*POINT (j-1,i-1))
 140 PRINT p$(LEN p$);
 150 NEXT j
 160 NEXT i
 170 PRINT #0;" Czy chcesz konty
nuowac ? (t/n)"
 180 LET k$=INKEY$: IF k$<>"t" A
ND k$<>"T" AND k$<>"n" AND k$<>"
N" THEN GO TO 180
 190 IF k$="t" OR k$="T" THEN GO
 SUB 210: GO TO 40
 200 GO SUB 210: STOP
 210 BORDER 7: PAPER 7: INK 0: C
LS : RETURN
```

```
RUN

Podaj tekst (max. 8 znakow)

spectrum

Kolor ramki ? (0-7) 7

Kolor tla ? (0-7) 7

Kolor tekstu ? (0-7) 0

spectrum

spectrum

Czy chcesz kontynuowac ? (t/n)
```

JEDNOSTKI CIŚNIENIA

PODSTAWOWĄ jednostką ciśnienia w układzie SI jest paskal. Szybkie przeliczenie wartości ciśnienia w innych skalach umożliwia program "JEDNOSTKI CIŚNIENIA".

```
  10 REM ************************
  20 REM * JEDNOSTKI CISNIENIA *
  30 REM ************************
  40 BORDER 1: PAPER 1: INK 7: C
LS : LET S=100000: GO SUB 300
  50 INPUT "PODAJ WIELKOSC CISNI
ENIA ";W
  60 PRINT #0;"PODAJ JEDNOSTKE M
IARY (ABCDEFGH)"
  70 PAUSE 0: LET J$=INKEY$: IF
J$="" THEN GO TO 70
  80 LET J=CODE J$: IF NOT ((J>6
4 AND J<73) OR (J>96 AND J<105))
 THEN RUN
  90 LET J=J-96: IF J<0 THEN LET
 J=J+32
 100 IF J=2 THEN LET W=W*100
 110 IF J=3 THEN LET W=W*S/1.019
7
 120 IF J=4 THEN LET W=W*S/.9869
 130 IF J=5 THEN LET W=W*S/750
 140 IF J=6 THEN LET W=W*S/10.19
7
 150 IF J=7 THEN LET W=W*S
 160 IF J=8 THEN LET W=W*101325/
14.69
 170 PRINT AT 2,16;W
 180 PRINT AT 4,16;W/100
 190 PRINT AT 6,16;W*1.0197/S
 200 PRINT AT 8,16;W*.9869/S
 210 PRINT AT 10,16;W*750/S
 220 PRINT AT 12,16;W*10.197/S
 230 PRINT AT 14,16;W/S
 240 PRINT AT 16,16;W*14.69/1013
25
 250 PRINT #0;AT 1,0;" CZY CHCE
SZ JESZCZE RAZ (T/N) ?"
 260 PAUSE 0: LET J$=INKEY$: IF
J$="T" OR J$="t" THEN RUN
 270 STOP
 300 PRINT "      JEDNOSTKI CISN
IENIA"
 310 PRINT '"A  paskal - Pa"
 320 PRINT '"B  hPa"
 330 PRINT '"C  at"
 340 PRINT '"D  atm"
 350 PRINT '"E  mmHg"
 360 PRINT '"F  m sl. wody"
 370 PRINT '"G  bar"
 380 PRINT '"H  lb./sq. in."
 390 RETURN
```

```
RUN
        JEDNOSTKI CISNIENIA

A  paskal - Pa   100000

B  hPa           1000

C  at            1.0197

D  atm           0.9869

E  mmHg          750

F  m sl. wody    10.197

G  bar           1

H  lb./sq. in.   14.497903
```

ŻÓŁW W BASICU

CHCĄC rysować na ekranie stosując grafikę żółwia w LOGO należy najpierw wczytać dość długi program LOGO, co nieraz może być bardzo kłopotliwe. Pewną namiastką LOGO może być program "ŻÓŁW W BASICU", który symuluje grafikę żółwia. Stosowanie tego programu może dać wyobrażenie o ruchu żółwia na ekranie, o sposobie tworzenia nawet skomplikowanych rysunków.

```
  10 REM ************************
  20 REM **** ZOLW W BASICU ****
  30 REM ************************
  40 LET AA=0: LET NX=0: LET NY=
0: LET Q=6: LET D=1: LET T=0: LE
T X=0: LET Y=0: GO SUB 700
  50 DIM A$(3): DIM A(3)
  60 FOR I=1 TO 3: READ A$(I): R
EAD A(I): NEXT I
  70 CLS : PLOT 128,84
  80 INPUT B$: IF B$="0" THEN ST
OP
  90 GO SUB 100: GO TO 80
 100 IF B$="" THEN RETURN
 110 LET N=1
 120 IF B$(1 TO 1)=A$(N) THEN GO
 TO 150
 130 LET N=N+1: IF N>3 THEN RETU
RN
 140 GO TO 120
 150 LET B$=B$(2 TO )
 160 FOR K=1 TO LEN B$: LET B=CO
DE (B$(K TO K)): IF B<48 OR B>57
 THEN RETURN
 170 NEXT K: LET V=VAL B$: GO SU
B A(N): RETURN
 300 LET V=V*D: LET V=T+V
 310 IF V>359 THEN LET V=V-360
 320 LET T=V: LET AA=V/(Q*30)*PI
 330 RETURN
 400 LET V=V*D: LET V=T-V
 410 IF V<0 THEN LET V=ABS V: LE
T V=360-V
 420 LET T=V: LET AA=V/(Q*30)*PI
 430 RETURN
 500 LET NX=V*SIN AA: LET NY=V*C
OS AA
 510 LET X=PEEK 23677: LET Y=PEE
K 23678: IF X+NX>255 OR X+NX<0 O
R Y+NY>175 OR Y+NY<0 THEN RETURN

 520 DRAW NX,NY: RETURN
 600 DATA "R",300,"L",400,"F",50
0
 700 PRINT TAB 12;"INSTRUKCJA"''
 710 PRINT "Program symuluje gra
fike zolwia."
 720 PRINT "Zolw moze sie porusz
ac do przodu"
 730 PRINT "skrecac w prawo oraz
 w lewo."''
 740 PRINT '"KOMENDY:"
 750 PRINT '"Fliczba - przesunie
cie zolwia do          przodu o
dana liczbe            krokow, n
p. F30"
 760 PRINT '"Rliczba - obrot zol
wia w prawo            o dana li
czbe stopni,           np. R60"
 770 PRINT '"Lliczba - obrot zol
wia w lewo             o dana li
czbe stopni,           np. L60"
 780 PRINT '"0         - koniec"
 790 PRINT #0;"   Nacisnij dowol
ny klawisz"
 800 PAUSE 0: RETURN

RUN
```

```
            INSTRUKCJA

Program symuluje grafike zolwia.
Zolw moze sie poruszac do przodu
skrecac w prawo oraz w lewo.

KOMENDY:

Fliczba - przesuniecie zolwia do
          przodu o dana liczbe
          krokow, np. F30

Rliczba - obrot zolwia w prawo
          o dana liczbe stopni,
          np. R60

Lliczba - obrot zolwia w lewo
          o dana liczbe stopni,
          np. L60

0       - koniec
   Nacisnij dowolny klawisz
```

NAJWIĘKSZY WSPÓLNY PODZIELNIK

TEN krótki program pozwala na szybkie obliczenie największego wspólnego podzielnika dwóch liczb. Po niewielkiej modyfikacji może być włączony jako moduł do większego programu realizującego skomplikowane obliczenia matematyczne.

```
  10 REM ************************
  20 REM * NAJWIEKSZY USPOLNY *
  30 REM *      PODZIELNIK      *
  40 REM ************************
  50 INPUT "PODAJ PIERWSZA LICZB
E ";A
  60 INPUT 'PODAJ DRUGA LICZBE"
,B
  70 LET C=A  LET D=B
  80 LET R=A-B*INT (A/B)
  90 IF R=0 THEN GO TO 120
 100 LET A=B  LET B=R
 110 GO TO 80
 120 PRINT "  NAJWIEKSZ USPOLNY
PODZIELNIK'
 130 PRINT '  LICZB    ",C,' i
,D
 140 PRINT '  WYNOSI   ",B

RUN

PODAJ PIERWSZA LICZBE 9876
PODAJ DRUGA LICZBE 346


   NAJWIEKSZ USPOLNY PODZIELNIK
   LICZB      9876 i 346
   WYNOSI     2
```

JEDNOSTKI MASY

JEDNOSTKĄ podstawową masy w układzie SI jest 1 kilogram, ale do mierzenia masy używa się także innych jednostek. Program "JEDNOSTKI MASY" pozwala na szybkie przeliczenie różnych jednostek masy.

```
  10 REM ***********************
  20 REM *** JEDNOSTKI  MASY ***
  30 REM ***********************
  40 BORDER 1: PAPER 1: INK 7: C
LS : GO SUB 400
  50 INPUT "PODAJ MASE ";W
  60 PRINT #0;"PODAJ JEDNOSTKE M
IARY"
  70 PAUSE 0: LET J$=INKEY$: IF
J$="" THEN GO TO 70
  80 LET J=CODE J$: IF NOT ((J>6
4 AND J<79) OR (J>96 AND J<111))
 THEN RUN
  90 LET J=J-96: IF J<0 THEN LET
 J=J+32
 100 IF J=2 THEN LET W=W*1000
 110 IF J=3 THEN LET W=W*100
 120 IF J=4 THEN LET W=W/1000
 130 IF J=5 THEN LET W=W/1E6
 140 IF J=6 THEN LET W=W*.0002
 150 IF J=7 THEN LET W=W*.031103
 160 IF J=8 THEN LET W=W*1016
 170 IF J=9 THEN LET W=W*50.8
 180 IF J=10 THEN LET W=W*.4536
 190 IF J=11 THEN LET W=W*.02835
 200 IF J=12 THEN LET W=W*.00172
2
 210 IF J=13 THEN LET W=W*907
 220 IF J=14 THEN LET W=W*45.36
 230 PRINT AT 2,20;W
 240 PRINT AT 3,20;W/1000
 250 PRINT AT 4,20;W/100
 260 PRINT AT 5,20;W*1000
 270 PRINT AT 6,20;W*1E6
 280 PRINT AT 7,20;W/.0002
 290 PRINT AT 8,20;W/.0311035
 300 PRINT AT 9,20;W/1016
 310 PRINT AT 10,20;W/50.8
 320 PRINT AT 11,20;W/.4536
 330 PRINT AT 12,20;W/.02835
 340 PRINT AT 13,20;W/.001722
 350 PRINT AT 14,20;W/907
 360 PRINT AT 15,20;W/45.36
 370 PRINT #0;AT 1,0;"  CZY CHCE
SZ JESZCZE RAZ (T/N) ?"
 380 PAUSE 0: LET J$=INKEY$: IF
J$="T" OR J$="t" THEN RUN
 390 STOP
 400 PRINT TAB 9;"JEDNOSTKI MASY
"
 410 PRINT '"A  kilogram - kg"
 420 PRINT "B  tona"
 430 PRINT "C  kwintal"
 440 PRINT "D  gram"
 450 PRINT "E  miligram"
 460 PRINT "F  karat jub."
 470 PRINT "G  uncja jub."
 480 PRINT "H  tona ang."
 490 PRINT "I  cetnar ang."
 500 PRINT "J  funt"
 510 PRINT "K  uncja"
 520 PRINT "L  drachma"
 530 PRINT "M  tona USA"
 540 PRINT "N  cetnar USA"
 550 RETURN
```

RUN

```
          JEDNOSTKI MASY

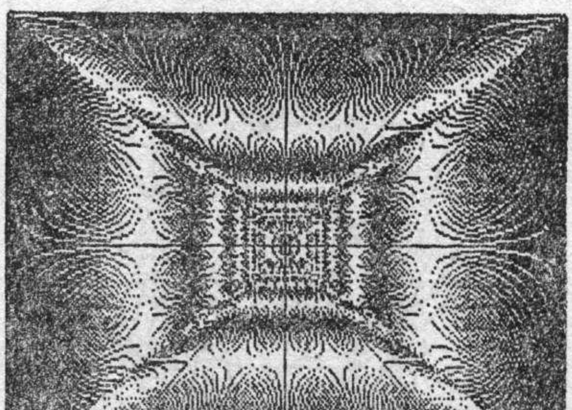
A  kilogram - kg     100
B  tona              0.1
C  kwintal           1
D  gram              100000
E  miligram          1E+8
F  karat jub.        500000
G  uncja jub.        3215.0723
H  tona ang.         .098425197
I  cetnar ang.       1.9685039
J  funt              220.45855
K  uncja             3527.3369
L  drachma           58072.009
M  tona USA          0.11025358
N  cetnar USA        2.2045855
```

MOZAIKA

ZWYKLE, aby otrzymać na ekranie ciekawą kompozycję graficzną trzeba napisać bardzo skomplikowany program. Bywają jednak krótkie programy, których wykonanie może dać zaskakujący efekt. Program "MOZAIKA" w kilku liniach tworzy obraz, który wydawać by się mogło nie jest łatwy do uzyskania.

```
  10 REM ***********************
  20 REM ******* MOZAIKA *******
  30 REM ***********************
  40 LET MX=127: LET MY=87: LET
SX=0: LET SY=0: OVER 1
  50 FOR K=0 TO 175: PLOT 0,K: D
RAW MX,MY-K: NEXT K
  60 FOR K=254 TO 0 STEP -1: PLO
T MX,MY+1: DRAW MX-K,MY: NEXT K
  70 FOR K=175 TO 0 STEP -1: PLO
T MX,MY: DRAW MX,K-MY: NEXT K
  80 FOR K=0 TO 254: PLOT MX,MY:
 DRAW MX-K,-MY: NEXT K
```

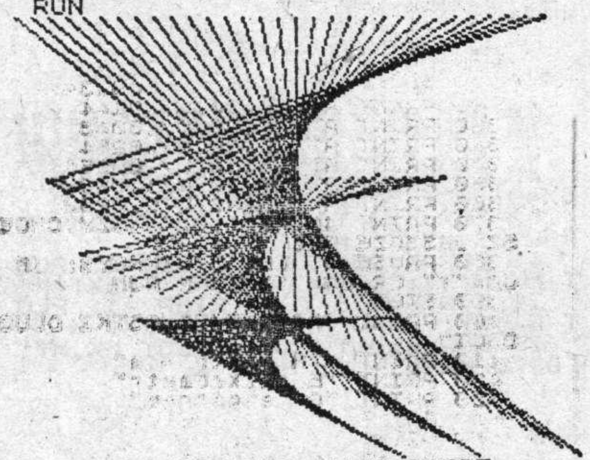
RUN



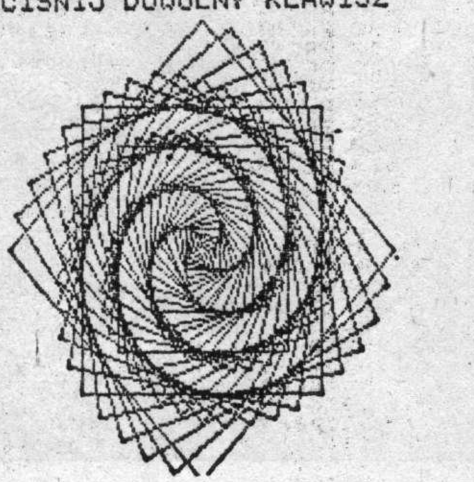
GRAFIKA

CIEKAWĄ kompozycję graficzną można uzyskać obracając jakąś figurę geometryczną. Program "GRAFIKA" pozwala wykreślić dwie kompozycje kreśląc odcinki oraz figurę utworzoną z odcinków — kwadrat. Zmiana parametrów w tym programie pozwala na dość ciekawe przekształcanie uzyskiwanych obrazów.

```
 10 REM ***********************
 20 REM ******* GRAFIKA *******
 30 REM ***********************
 40 BORDER 1: PAPER 1: INK 7: C
LS
 50 FOR I=0 TO 247 STEP 8
 60 LET X1=I*3/4: LET X2=I/2
 70 LET M1=110/255*I
 80 LET M2=86/200*X1
 90 LET M3=55/127*X2
 100 PLOT 255-I,M1: DRAW 2*I-255
,165-M1
 110 PLOT 210-X1,M2: DRAW 2*X1-1
95,105-M2
 120 PLOT 176-X2,M3: DRAW 2*X2-1
20,52-M3
 130 NEXT I
 140 PRINT #0;"   NACISNIJ DOWOL
NY KLAWISZ": PAUSE 0
 150 CLS : LET X0=127: LET Y0=87
 160 FOR I=360 TO 1 STEP -9
 170 LET B=.87*I/3.6
 180 LET SI=B*COS (I*PI/180): LE
T X=X0+SI
 190 LET CI=B*SIN (I*PI/180): LE
T Y=Y0+CI
 200 PLOT X,Y: DRAW X0+CI-X,Y0-S
I-Y
 210 DRAW -SI-CI,SI-CI
 220 DRAW SI-CI,SI+CI
 230 DRAW X-X0+CI,Y-Y0-SI
 240 NEXT I
 RUN
```



NACISNIJ DOWOLNY KLAWISZ



SYSTEMY LICZBOWE

LICZBY wewnątrz komputera przechowywane są w postaci ciągu zer i jedynek. W odniesieniu do tak zapisanych liczb mówimy. że są to liczby dwójkowe (binarne) lub inaczej, że są zapisane w dwójkowym (binarnym) systemie liczbowym. Określenie systemu dwójkowy bierze się od przyjętej podstawy liczenia, która równa jest 2, tzn. wyróżnione są dwa stany do reprezentacji liczb. Oprócz systemu dwójkowego w informatyce stosowany jest m.in. system szesnastkowy (heksalny) oraz ósemkowy (oktalny) których podstawami są odpowiednio 16 (szesnaście wyróżnionych stanów 0, ..., 1, A, ... F) i 8 (osiem wyróżnionych stanów 0, 1, ..., 7).

W życiu codziennym posługujemy się systemem dziesiętnym (dziesięć stanów wyróżnionych — cyfry od 0 do 9). Mamy wobec tego trudności z ocenieniem "ile to jest" 3A5C zapisane szesnastkowo w systemie dziesiętnym. Poniższy program umożliwia zamianę liczby zapisanej w jednym systemie liczbowym na odpowiednią postać w innym systemie. Zamieniane liczby mogą mieć podstawę od 2 do 36.

```
 10 REM ***********************
 20 REM ** SYSTEMY LICZBOWE ***
 30 REM ***********************
 40 INPUT "Zamieniana liczba :"
;a$
 50 INPUT "z podstawy ";p1
 60 IF p1<2 OR p1>36 THEN PRINT
#0;"Tylko od 2 do 36": GO TO 50
 70 INPUT "na podstawe ";p2
 80 IF p2<2 OR p2>36 THEN PRINT
#0;"Tylko od 2 do 36": GO TO 70
 90 LET t=0
 100 FOR c=1 TO LEN a$
 110 LET v=CODE a$(c)
 120 IF v>47 AND v<58 THEN LET v
1=v-48
 130 IF v>64 AND v<91 THEN LET v
1=v-55
 140 IF v>96 AND v<123 THEN LET
v1=v-87
 150 IF v1>=p1 THEN PRINT #0;":l
a cyfra (litera) w liczbie !": P
AUSE 200: GO TO 40
 160 LET t=t*p1+v1
 170 NEXT c
 180 LET b$=""
 190 IF t=0 THEN GO TO 260
 200 LET v1=t-INT (t/p2)*p2
 210 LET t=(t-v1)/p2
 220 IF v1<10 THEN LET v=v1+48
 230 IF v1>9 THEN LET v=v1+55
 240 LET b$=CHR$ v+b$
 250 GO TO 190
 260 PRINT AT 4,0;"Liczba......
";a$''"o podstawie liczenia...."
;p1''"ma przy podstawie......";
p2''"postac...... ";b$
 270 PRINT #0;" Czy chcesz konty
nuowac ? (t/n)"
 280 LET k$=INKEY$: IF k$<>"t" A
ND k$<>"T" AND k$<>"n" AND k$<>"
N" THEN GO TO 280
 290 IF k$="t" OR k$="T" THEN CL
S : GO TO 40
 300 STOP

 RUN

 Zamieniana liczba :4FFF

 z podstawy 16

 na podstawe 2

 Liczba...... 4FFF

 o podstawie liczenia... 16

 ma przy podstawie...... 2

 postac...... 100111111111111

 Czy chcesz kontynuowac ? (t/n)
```

KURSY WALUT

PROGRAM "KURSY WALUT" pozwala na szybkie przeliczanie jednych walut na inne wg wcześniej podanej relacji. Niewielka modyfikacja programu pozwala wykorzystać go do przeliczania innych wielkości, których relacje zmieniają się lub są stałe.

```
  10 REM ***********************
  20 REM ***** KURSY WALUT *****
  30 REM ***********************
  40 LET IW=10: DIM K$(IW,10): D
IM W$(IW,8): DIM K(IW): DIM W(IW
)
  50 FOR I=1 TO IW: READ K$(I):
READ W$(I): NEXT I
  60 FOR I=1 TO IW: LET W(I)=0:
NEXT I: GO SUB 400: LET K(1)=1
  70 FOR I=2 TO IW
  80 PRINT #0;"1 US$ = X * ";W$(
I);K$(I);"  PODAJ WARTOSC X, CZY
LI KURS";: PAUSE 0: INPUT "X=";K
(I): IF K(I)<=0 THEN GO TO 80
  90 NEXT I
 100 INPUT "PODAJ ILOSC WALUTY "
;WW
 110 PRINT #0;"PODAJ JEDNOSTKE W
ALUTY"
 120 LET A$=INKEY$: IF A$="" THE
N GO TO 120
 130 LET A=CODE A$: IF A<65 OR A
>106 OR (A>74 AND A<97) THEN GO
TO 120
 140 LET A=A-96: IF A<0 THEN LET
 A=A+32
 150 LET W(1)=WW/K(A): GO SUB 30
0: GO SUB 400: GO TO 100
 300 FOR I=1 TO IW: LET W(I)=K(I
)*W(1): NEXT I: RETURN
 400 CLS : PRINT TAB 10;"KURSY W
ALUT"
 410 FOR I=1 TO IW: LET W=2*I: P
RINT AT W,1;CHR$ (I+64);AT W,3;K
$(I);AT W,14;W$(I);AT W,22;W(I);
: NEXT I
 420 RETURN
 500 DATA "USA","DOLAR","RFN","M
ARKA","FRANCJA","FRANK","W.BRYTA
NIA","FUNT","HISZPANIA","PESETA"
 510 DATA "SZWECJA","KORONA","WL
OCHY","LIR","AUSTRIA","SZYLING",
"KANADA","DOLAR","POLSKA","ZLOTY
"
RUN
            KURSY WALUT

 A USA          DOLAR    1

 B RFN          MARKA    2

 C FRANCJA      FRANK    6

 D W.BRYTANIA   FUNT     0.8

 E HISZPANIA    PESETA   150

 F SZWECJA      KORONA   7

 G WLOCHY       LIR      1700

 H AUSTRIA      SZYLING  8

 I KANADA       DOLAR    1.2

 J POLSKA       ZLOTY    650

PODAJ ILOSC WALUTY 1
PODAJ JEDNOSTKE WALUTY A
```

JEDNOSTKI DŁUGOŚCI

JEDNOSTKĄ podstawową długości w układzie SI jest 1 metr, jednak w praktyce spotyka się również inne miary. Program "JEDNOSTKI DŁUGOŚCI" pozwala na szybkie przeliczenie różnych miar długości.

```
  10 REM ***********************
  20 REM * JEDNOSTKI  DLUGOSCI *
  30 REM ***********************
  40 BORDER 1: PAPER 1: INK 7: C
LS : GO SUB 400
  50 INPUT "PODAJ DLUGOSC ";W
  60 PRINT #0;"PODAJ JEDNOSTKE M
IARY"
  70 PAUSE 0: LET J$=INKEY$: IF
J$="" THEN GO TO 70
  80 LET J=CODE J$: IF NOT ((J>6
4 AND J<79) OR (J>96 AND J<111))
 THEN RUN
  90 LET J=J-96: IF J<0 THEN LET
 J=J+32
 100 IF J=2 THEN LET W=W/1E6
 110 IF J=3 THEN LET W=W/1E10
 120 IF J=4 THEN LET W=W*1852
 130 IF J=5 THEN LET W=W*185.2
 140 IF J=6 THEN LET W=W*8534
 150 IF J=7 THEN LET W=W*1067
 160 IF J=8 THEN LET W=W*1609.34
 170 IF J=9 THEN LET W=W*.9144
 180 IF J=10 THEN LET W=W*.3048
 190 IF J=11 THEN LET W=W*.0254
 200 IF J=12 THEN LET W=W*.00254
 210 IF J=13 THEN LET W=W*1.8288
 220 IF J=14 THEN LET W=W*5.0292
 230 PRINT AT 2,20;W
 240 PRINT AT 3,20;W*1E6
 250 PRINT AT 4,20;W*1E10
 260 PRINT AT 5,20;W/1852
 270 PRINT AT 6,20;W/185.2
 280 PRINT AT 7,20;W/8534
 290 PRINT AT 8,20;W/1067
 300 PRINT AT 9,20;W/1609.34
 310 PRINT AT 10,20;W/.9144
 320 PRINT AT 11,20;W/.3048
 330 PRINT AT 12,20;W/.0254
 340 PRINT AT 13,20;W/.00254
 350 PRINT AT 14,20;W/1.8288
 360 PRINT AT 15,20;W/5.0292
 370 PRINT #0;AT 1,0;"  CZY CHCE
SZ JESZCZE RAZ (T/N) ?"
 380 PAUSE 0: LET J$=INKEY$: IF
J$="T" OR J$="t" THEN RUN
 390 STOP
 400 PRINT TAB 9;"JEDNOSTKI DLUG
OSCI"
 410 PRINT "A  metr - m"
 420 PRINT "B  mikrometr"
 430 PRINT "C  angstrem"
 440 PRINT "D  mila morska"
 450 PRINT "E  kabel"
 460 PRINT "F  mila polska"
 470 PRINT "G  wiorsta rosyjska"
 480 PRINT "H  mila angielska"
 490 PRINT "I  jard"
 500 PRINT "J  stopa"
 510 PRINT "K  cal"
 520 PRINT "L  linia"
 530 PRINT "M  sazen"
 540 PRINT "N  pret"
 550 RETURN
RUN
         JEDNOSTKI DLUGOSCI

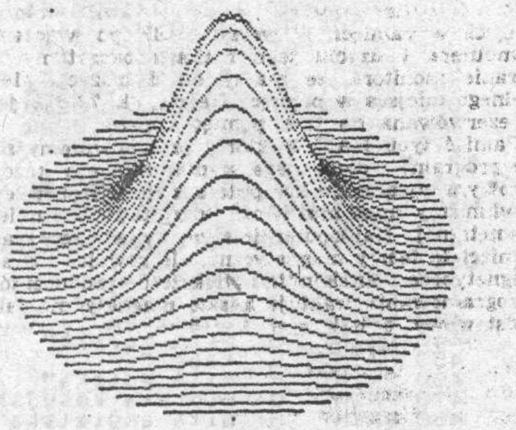
A  metr - m             8534
B  mikrometr            8.534E+9
C  angstrem             8.534E+13
D  mila morska          4.6079914
E  kabel                46.079914
F  mila polska          1
G  wiorsta rosyjska     7.9981256
H  mila angielska       5.3027949
I  jard                 9332.8959
J  stopa                27998.688
K  cal                  335984.25
L  linia                3359842.5
M  sazen                4666.4479
N  pret                 1696.8902
```

POWIERZCHNIE

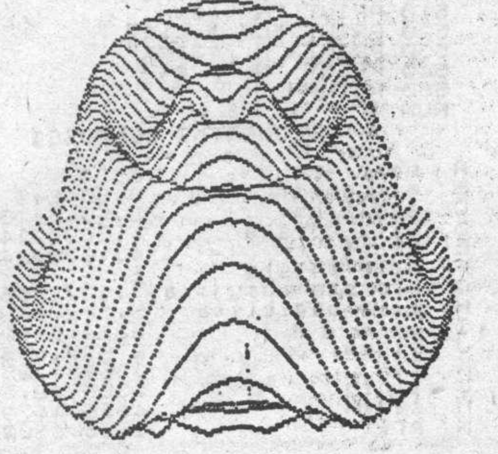
W BASIC-u można otrzymać również efektywną grafikę przestrzenną. Program "POWIERZCHNIE" wykorzystując funkcje zdefiniowane przez programistę oraz funkcje standardowe EXP, SIN, SQR, INT kreśli obraz skomplikowanych funkcji trzech zmiennych. Duża ilość obliczeń powoduje długi czas wykonania programu.

```
  10 REM ************************
  20 REM ***** POWIERZCHNIE ****
  30 REM ************************
  40 DEF FN A(Z)=90*EXP (-Z*Z/15
00)
  50 DEF FN C(Z)=38*(SIN (Z/24)+
.48*SIN (3*Z/24))+20
  60 FOR A=1 TO 2: CLS
  70 LET G=128: LET K=40: LET B=
10000
  80 FOR X=-100 TO 0
  90 LET L=-60: LET H=5*INT (SQR
 (B-X*X)/5)
 100 FOR Y=H TO -H STEP -5
 110 IF A=1 THEN GO SUB 180
 120 IF A=2 THEN GO SUB 190
 130 IF Z>L THEN LET L=Z: PLOT G
+X,Z+K: PLOT G-X,Z+K
 140 NEXT Y: NEXT X
 150 PRINT #0;"    NACISNIJ DOWO
LNY KLAWISZ"
 160 PAUSE 0: NEXT A
 170 STOP
 180 LET Z=25+FN A(SQR (X*X+Y*Y)
)-.6*Y: RETURN
 190 LET Z=25+FN C(SQR (X*X+Y*Y)
)-.6*Y: RETURN

RUN
```



NACISNIJ DOWOLNY KLAWISZ



SKALE TERMOMETRYCZNE

JEDNOSTKĄ podstawową temperatury w układzie SI jest stopień Kelvina, jednak w życiu codziennym spotykamy się także z innymi skalami temperatur. Na szybkie przeliczanie wartości temperatur w różnych skalach pozwala program "SKALE TERMOMETRYCZNE".

```
  10 REM ************************
  20 REM  SKALE TERMOMETRYCZNE
  30 REM ************************
  40 GO SUB 300
  50 INPUT "PODAJ WARTOSC TEMPER
ATURY ";T
  60 PRINT #0;AT 1,0;"PODAJ W JA
KIEJ SKALI (C/K/F/R) "
  70 LET S$=INKEY$: IF S$="" OR
(S$<>"C" AND S$<>"K" AND S$<>"F"
 AND S$<>"R" AND S$<>"c" AND S$<
>"k" AND S$<>"f" AND S$<>"r") TH
EN GO TO 70
  80 IF S$="C" OR S$="c" THEN LE
T C=T
  90 IF S$="K" OR S$="k" THEN LE
T C=T-273.15
 100 IF S$="F" OR S$="f" THEN LE
T C=(T-32)*5/9
 110 IF S$="R" OR S$="r" THEN LE
T C=T/.8
 120 LET K=C+273.15
 130 LET F=(C*9)/5+32
 140 LET R=C*.8
 150 LET S$="SKALA ": CLS
 160 PRINT S$;"CELSJUSZA   ";C
 170 PRINT S$;"KELVINA     ";K
 180 PRINT S$;"FAHRENHEITA ";F
 190 PRINT S$;"REAUMURA    ";R
 200 PRINT #0;AT 1,0;" CZY CHCE
SZ JESZCZE RAZ (T/N) ?"
 210 PAUSE 0: LET S$=INKEY$: IF
S$="T" OR S$="t" THEN RUN
 220 STOP
 300 PRINT AT 0,6;"SKALE TERMOME
TRYCZNE"
 310 PRINT '"Program pozwala na
obliczanie "
 320 PRINT "wartosci temperatur
w roznych"
 330 PRINT "skalach.Temperatura
moze byc"
 340 PRINT "podana w skali :"
 350 PRINT AT 7,17;"CELSJUSZA"
 360 PRINT AT 9,17;"KELVINA"
 370 PRINT AT 11,17;"FAHRENHEITA
"
 380 PRINT AT 13,17;"REAUMURA"
 390 RETURN
RUN
```

```
      SKALE TERMOMETRYCZNE

Program pozwala na obliczanie
wartosci temperatur w roznych
skalach.Temperatura moze byc
podana w skali :

                 CELSJUSZA

                 KELVINA

                 FAHRENHEITA

                 REAUMURA


SKALA CELSJUSZA    37.777778
SKALA KELVINA      310.92778
SKALA FAHRENHEITA  100
SKALA REAUMURA     30.222222

SKALA CELSJUSZA    0
SKALA KELVINA      273.15
SKALA FAHRENHEITA  32
SKALA REAUMURA     0
```

7 Zasady wyboru i eksploatacji mikrokomputera

ZAKUP mikrokomputera związany jest z realizacją określonych, ważnych potrzeb przyszłego użytkownika, z nakładem finansowym nie zawsze „mikro", ze spodziewanymi efektami. Dlatego też przedsięwzięcie to powinno być odpowiednio rozpoznane, przygotowane, zaplanowane i zrealizowane. Oczywiście inaczej do tego zagadnienia podejdzie amator domowej informatyki, a inaczej specjalista odpowiedzialny za wykorzystanie mikrokomputera profesjonalnego do zarządzania w przedsiębiorstwie. Ten ostatni przy wyborze mikrokomputera powinien pamiętać o pewnych zasadach oraz przestrogach. Zakup mikrokomputera jest inwestycją kosztowną. Dlatego też nie należy dokonywać zakupów w pośpiechu, bez zasięgnięcia fachowych porad. Należy więc dokładnie zdefiniować swoje potrzeby i oczekiwania, cele i obszary zastosowań, aby do nich właśnie dobrać odpowiedni sprzęt i oprogramowanie. Bardzo często zdarza się, że do trudnych zagadnień zakupuje się prosty sprzęt, który nie jest w stanie sprostać potrzebom. Powstała wówczas sytuacja powoduje negatywne opinie o korzyściach z zastosowań informatyki. Należy więc zbierać informacje o producentach, sprzęcie i oprogramowaniu, brać udział w targach, giełdach i pokazach sprzętu i oprogramowania, posiłkować się radami użytkowników, doradców i ekspertów. Wybór mikrokomputera jest też określonym procesem przetwarzania danych, w wyniku którego powstaje koncepcja rozwiązania problemu. Urządzenia oferowane przez różne firmy są tak różnorodne, że bez problemu można je dostosować do indywidualnych potrzeb użytkowników. Nie należy zapominać o tym, że każdy sprzęt jest mniej lub bardziej zawodny, a ciągłość eksploatacji może zapewnić sprawny serwis techniczny. Eksploatując system powinniśmy dążyć do tego, aby w pełni wykorzystać jego możliwości. Dlatego też należy poddawać częstej analizie przydatność istniejących rozwiązań, wytyczać kierunki rozwoju sprzętu i systemu — system, który nie rozwija się, starzeje się (umiera).

Powyższe zasady tylko w niewielkim stopniu odnoszą się do amatorów domowej informatyki i nieprofesjonalnych zastosowań. Przyszli użytkownicy zadają sobie pytanie: jaki komputer kupić i jakimi kryteriami kierować się przy jego wyborze? Natomiast większość posiadaczy tej klasy sprzętu po fascynacji wspaniałą zabawą, jaką zapewniają gry komputerowe oczekuje wskazówek i pomocy jak „mądrzej" użytkować swój mikrokomputer.

Obecnie na świecie produkowanych i dostępnych na rynku jest kilkadziesiąt typów mikrokomputerów. Wśród gamy różnorodnych urządzeń poszukujemy dla siebie mikrokomputera taniego, popularnego i najsprawniejszego. Mikrokomputery domowe stanowią najtańszą grupę komputerów. Na ogół jest to zakup kosztowny, ale ze wszech miar słuszny. Pozwala bowiem efektywniej pracować, zapewnia znakomitą zabawę. Najbardziej popularne i dostępne mikrokomputery dysponują bogatym oprogramowaniem. Mogą to być zarówno gry jak i programy edukacyjne. Sprawność komputera można opisać kilkoma najistotniejszymi parametrami. Podstawowym elementem jest mikroprocesor. Decyduje on o długości słowa maszynowego oraz szybkości działania mikrokomputera. Obecnie powszechnie stosuje się w komputerach domowych mikroprocesor 8-bitowy (np. w ZX SPECTRUM mikroprocesor Z80A). Oczywiście nowocześniejsze to mikroprocesory 16-bitowe, ale już w niedalekiej przyszłości w powszechnym użytkowaniu będą mikrokomputery domowe wyposażone w bardzo szybkie 32-bitowe mikroprocesory. Im szybszy procesor tym na ogół sprawniejszy komputer. Szybkość obliczeń komputera najczęściej wyrażona jest ilością wykonywanych operacji w ciągu sekundy. Rzadziej producent określa ten parametr jako częstotliwość zegara: im wyższa tym sprawniejszy komputer. Każdy mikrokomputer wyposażony jest w dwa typy pamięci: stałą ROM oraz dostępną dla użytkownika, dla jego programów i danych, zwaną RAM. Im większa pojemność tej pamięci tym mikrokomputer jest wygodniejszy w użyciu. Niektórzy producenci umożliwiają użytkownikom rozszerzanie pamięci RAM „domowym" sposobem. Użytkownik ZX SPECTRUM po napisaniu i uruchomieniu instrukcji "PRINT 65535 — USR 7962" otrzyma na ekranie monitora informację ile pozostaje w danej chwili wolnego miejsca w pamięci. W wersji z 48K po włączeniu komputera i użyciu tego rozkazu odczytamy na ekranie monitora, że mamy do dyspozycji 41472 wolnego miejsca w pamięci. Reszta, ok. 7,5 KB jest zarezerwowana na tzw. pamięć obrazu.

Pamięć typu RAM w której przechowujemy nasze programy i dane ulega wymazaniu przy każdorazowym wyłączeniu komputera z sieci. Wobec tego powinniśmy mieć możliwość korzystania z pamięci zewnętrznej. Najpopularniejszym typem tej klasy pamięci to taśmy magnetyczne, ale również i dyski magnetyczne. Znakomita większość producentów oprogramowania traktuje kasetę magnetyczną jako podstawowy nośnik gier i programów edukacyjnych. Zaletą tej klasy pamięci zewnętrznej jest jej niska cena natomiast wadą — długi czas oczekiwania na zakończenie operacji zapisu lub odczytu programu. Nierzadko użytkownik musi wykazać dużo cierpliwości podczas tych operacji aby osiągnąć sukces. Oczywiście prawidłowa eksploatacja magnetofonu i kaset z programami może znacznie poprawić efekty współpracy mikrokomputera z taśmową pamięcią zewnętrzną. Przede wszystkim należy nagrywać i odtwarzać programy na tym samym magnetofonie. Dbać o czystość kaset i magnetofonu. Kasety po wykorzystaniu zawsze przewinąć do początku lub końca. Nie nagrywać na kasetę zbyt wielu programów: 2—3 z każdej strony. Stanowisko na którym pracujemy z komputerem powinno zawsze lśnić czystością, nasze ręce również. Pracujemy spokojnie z rozwagą i cierpliwością.

Droższe mikrokomputery dysponują specjalnymi magnetofonami dostarczonymi przez producenta wraz ze sprzętem. Pamięcią pośrednią między taśmą magnetyczną a stacją dysków są tzw. mikrodrive'y, czyli pętle taśmy magnetycznej bez końca. Ale najwygodniejszą, niezawodną, o krótkim czasie dostępu do programów i danych jest pamięć zewnętrzna na dyskach magnetycznych. Niestety stacje dysków do mikrokomputera są bardzo drogie,

przekraczające kilkakrotnie cenę samego mikrokomputera.

Domowe mikrokomputery zaopatrzone są w klawiatury różnego typu. Mogą to być urządzenia z klawiszami gumowymi i plastikowymi. Oczywiście korzystniejsze są klawiatury solidne z tworzywa sztucznego. Niebagatelne znaczenie ma również ten typ klawiatury, który przy małej liczbie klawiszy dysponuje bogatymi możliwościami, np. za dotknięciem klawisza wywoływane są całe słowa kluczowe lub polecenia dla maszyny (np. ZX SPECTRUM dysponuje 40 klawiszami za pomocą których wywołujemy 180 różnych funkcji).

Dobry komputer winien mieć również syntetyzatory dźwięku sterowane z klawiatury oraz bogate możliwości graficzne, o rozdzielczości co najmniej 256×176 punktów na ekranie monitora. Pamiętać należy również o tym, że kolor uzyskamy na ekranie telewizora, który ma możliwość pracy w systemie PAL oraz dysponuje 36 kanałem. Do pełnej konfiguracji brak naszemu mikrokomputerowi już tylko drukarki, aby wyprowadzić instrukcje programu lub wyniki obliczeń na papier jako tzw. twardą kopię. Niestety, podobnie jak stacje dysków, drukarki są również bardzo drogie. Gdzie można kupić mikrokomputer? Najlepiej za granicą, bo wówczas zapłacimy taniej i będziemy mieli większy wybór. W kraju za złotówki mikrokomputery oferują firmy polonijne, komisy i skupy.

Aby uniknąć rozczarowań już przy pierwszym włączeniu urządzenia do sieci, powinniśmy pamiętać o właściwym obchodzeniu się z tym delikatnym sprzętem. Wszelkiego rodzaju instrukcje obsługi zawierają ostrzeżenia i przestrogi. A ponieważ w większości są to instrukcje obcojęzyczne, przypomnijmy kilka podstawowych zasad pracy z mikrokomputerem:

- przyłączania i wyłączania dodatkowych urządzeń poprzez złącze krawędziowe wykonywać należy tylko przy wyłączonym komputerze,
- zasilacz najpierw włączamy do sieci, a dopiero później podłączamy do niego komputer,
- przed przyłączeniem czegokolwiek do złącza krawędziowego należy najpierw upewnić się, czy wtyk wyposażony jest w plastikową wstawkę, tzw. splot,
- unikajmy przegrzewania komputera,
- trzymajmy komputer w czystości, po skończonej pracy chowamy go do pudełka,
- w czasie pracy nie ustawiamy mikrokomputera na miękkim podłożu, ponieważ w ten sposób utrudniamy dostęp powietrza do otworów wentylacyjnych,
- jeśli często bawimy się grami zręcznościowymi, lepiej zaopatrzyć się w manipulator (joystick), bo grozić nam będzie wymiana klawiatury.

Na świecie w użytkowaniu jest kilkadziesiąt milionów mikrokomputerów dziesiątków typów, dla których napisano tysiące różnych programów. Zdecydowaną większość wśród nich stanowią gry. Dlatego też wielu ludzi traktuje mikrokomputer domowy wyłącznie jako zabawkę. A przecież u bardziej zaawansowanych użytkowników gry tracą na znaczeniu na korzyść oprogramowania edukacyjnego. Mikrokomputer może być pomocny do samodzielnego zdobywania wiedzy w szkole i w domu niezależnie od poziomu wiedzy użytkownika i kategorii wieku. Przy jego pomocy można nauczyć się programowania w językach: LOGO, PASCAL, ASSEMBLER ewentualnie BASIC. Programy edukacyjne mogą uczyć rozwiązywania skomplikowanych dla dziecka wyrażeń i działań arytmetycznych, służyć nauczaniu początkowemu dzieci, uczyć gramatyki, ortografii, języków obcych. Służą znakomitą pomocą dla ludzi niepełnosprawnych, niewidomych i niewidzących, wykorzystywane są dla korekty wad wymowy ludzi głuchych. W szkole na lekcjach fizyki, chemii i biologii, przy pomocy programów edukacyjnych można demonstrować przebieg zjawisk i procesów. Na lekcjach historii na zmieniających się na ekranie monitora mapach poznawać można dzieje narodów. Przy wykorzystaniu możliwości graficznych, kolorów i dźwięków i przy dużej inwencji twórcy programu, efekt musi być znakomity. Jaką się musi mieć satysfakcję, gdy jest się autorem takiego lub podobnego programu. Spróbuj więc i ty.

KIERUNKI ROZWOJU

Rozwój mikrokomputerów (zarówno środków technicznych, jak i oprogramowania) następuje w sposób niezwykle gwałtowny. Co ciekawsze i pocieszające jest on wyraźnie zauważalny również w kraju. Minął już bezpowrotnie okres produkcji w Polsce profesjonalnych mikrokomputerów 8-bitowych; na niespotykaną dotąd skalę, bo obejmującą tysiące sztuk, rozwinęła się produkcja mikrokomputerów 16-bitowych (standardem światowym stał się tu mikrokomputer IBM PC). Firmy mikrokomputerowe przygotowują się już do produkcji mikrokomputerów 32-bitowych. Można zatem powiedzieć, że opóźnienie nasze w możliwościach dostępu do najnowszego sprzętu jest nieznaczne (poprzez dostęp do elementów elektronicznych przodujących światowych producentów). Jest to jeszcze jednocześnie okres pozyskiwania sprzętu oraz rozpoznawanie oprogramowania narzędziowego. Wraz z masowym pojawieniem się sprzętu zaczyna pojawiać się bariera oprogramowania użytkowego, którą musimy pokonać własnymi siłami ze względu na specyfikę funkcjonowania naszej gospodarki i związanych z nim systemów zarządzania przedsiębiorstwami.

Rozwój mikrokomputerów dokonuje się pod wpływem dążenia do realizacji następujących celów:

- obniżenia kosztów produkcji i eksploatacji środków technicznych, przy jednoczesnym wzroście niezawodności sprzętu,
- wzrostu wydajności i zasobów sprzętu,
- potrzeby dostosowania środków informatyki do specyfiki bezpośrednich zastosowań (potrzeb użytkowników),
- zapewnienia łatwości i prostoty obsługi i eksploatacji sprzętu.

Tendencja pierwsza przejawia się między innymi w dziesięciokrotnej lub większej obniżce cen poszczególnych elementów elektronicznych w okresach dziesięcioletnich, nowych technologiach produkcji (oraz większej skali integracji układów scalonych), wykorzystania nowych materiałów (np. zamiast krzemu arsenek galu).

Tendencję drugą obrazują między innymi mikroprocesory o coraz dłuższym słowie (powszechne w zastosowaniu staną się w najbliższych już latach mikroprocesory 32-bitowe), pojemniejsze pamięci zewnętrzne i wewnętrzne, nowe rodzaje pamięci (np. dyski optyczne), nowe urządzenia wejścia-wyjścia.

Zastosowanie mikrokomputerów powstaje zwykle w wyniku wyraźnego zapotrzebowania użytkownika: inżyniera, projektanta, księgowego, magazyniera, nauczyciela itp. spełniając jego specyficzne wymagania. Stąd dająca się zaobserwować różnorodność zastosowań i funkcjonujących w nich mikrokomputerów jako automatycznych stanowisk pracy.

Kolejne fazy rozwoju tych zastosowań to systemy wielostanowiskowe, łączenie mikrokomputerów w sieci oraz podłączanie ich do sieci komputerowych.

8 SŁOWNIK PODSTAWOWYCH TERMINÓW

adres — symbol (słowo) identyfikujący komórkę lub obszar pamięci lub też zespół funkcjonalny; nazwy adresów pochodzą od metod adresowania,

algorytm — dokładny przepis wykonania szeregu operacji w celu rozwiązania określonego zadania,

architektura mikroprocesora — określa współpracę grup funkcjonalnych mikroprocesora tj. jednostki sterującej i arytmometra, ich budowę, sposób działania, właściwości oraz powiązania między nimi,

arytmometr — część jednostki centralnej względnie mikroprocesora przeznaczona do wykonywania operacji arytmetycznych i logicznych,

assembler — zorientowany maszynowo symboliczny język programowania, w którym rozkazy i większość adresów oparta jest na kodzie mnemotechnicznym, jest to język z grupy języków niskiego poziomu programowania (zorientowany sprzętowo),

BASIC — Beginners All Purpose Symbolic Instruction Code, bardzo prosty, łatwy do nauki, problemowy język programowania,

bajt — umowna jednostka przetwarzania informacji w organizacji komputerów. Obejmuje ona 8 bitów. Z tego 7 bitów informacyjnych i jeden bit kontrolny; wielokrotność tej jednostki to: Kbajt (kilobajt) = 1024 bajtów i Mbajt (megabajt) = 1024×1024 bajtów. Skróty jednostek: KB i MB,

bit — cyfra w zapisie dwójkowym,

dysk elastyczny — dyskietka, tzw. miękkie dyski, odmiana dysku magnetycznego. Przy małych rozmiarach charakteryzuje się dużą pojemnością. Stosuje się dyski o średnicy: 8", 5¼", 3" (1" = 1 cal),

edytor — program redagujący, umożliwia redagowanie przez monitor ekranowy,

hardware — sprzęt komputerowy,

informatyka — nauka o systemie przetwarzania informacji,

instrukcja — przepis działania sformułowany w języku programowania,

interface — złącze,

interpreter — język, który w sposób kroczący (sekwencyjny) interpretuje instrukcje programu napisanego w języku innym niż język komputera.

Interpreter tym różni się między innymi od kompilatora i translatora, że nie tworzy nowego pośredniego programu (programu wynikowego), a interpretowane instrukcje są realizowane bezpośrednio przez komputer,

jednostka centralna — CPU (Central Processing Unit) — procesor centralny — część komputera na którą składa się jednostka sterująca, arytmometr, urządzenia WE/WY i pamięć operacyjna, a w której dokonuje się operacji logiczno-arytmetycznych oraz koordynuje i steruje działaniem innych jednostek,

kompatybilność — wymienność,

kompilator — translator programu opracowanego w języku problemowo-zorientowanym tworzącym program w kodzie komputera,

komputer — zestaw urządzeń lub urządzenie o programowalnej pamięci do przetwarzania danych, które po wprowadzeniu do pamięci programu oraz danych przetwarza je wg programu w sposób automatyczny,

konfiguracja — pod pojęciem tym rozumie się zestaw (układ) sprzętu komputerowego (hardware do określonych zadań, w odniesieniu do mikrokomputerów, konfiguracja wynika z ich architektury,

kontroler — sterownik, programator,

lista rozkazów — spis repertuaru znaków wraz z ich opisem,

manipulator — joystick, urządzenie do ręcznego manipulowania,

mikrokomputer — układ scalony lub ich zestaw, który pod względem funkcjonalnym odpowiada komputerowi, zawiera on jednostkę centralną w postaci mikroprocesora, pamięć programu i danych, system wejścia, wyjścia oraz (coraz powszechniej) urządzenia peryferyjne w postaci drukarek, monitorów, itp.,

mikroprocesor — układ scalony, który pod względem funkcjonalnym porównywalny jest z jednostką centralną komputera, zawiera on jednostkę sterującą i arytmometr,

monitor ekranowy — urządzenie peryferyjne do wprowadzania i wyprowadzania danych za pomocą ekranu i klawiatury,

pamięć operacyjna — pamięć typu zapis/odczyt o dostępie swobodnym (bezpośrednim) do programów, wyników pośrednich, ewentualnie danych,

pamięć zewnętrzna — każda pamięć, która nie jest pamięcią operacyjną,

ploter — urządzenie rysujące,

RAM — pamięć w której przechowuje się wyłącznie dane,

ROM — pamięć programu w której przechowuje się wyłącznie programy,

wejściowy — związany z wprowadzaniem danych,

wyjściowy — związany z wyprowadzaniem danych,

znak — pojedynczy element umownego zbioru symboli używanych w systemie przetwarzania danych.

Wydawca: Szczecińskie Wydawnictwo Prasowe — redakcja „Kurier Szczeciński" RSW „Prasa-Książka-Ruch".
Opracowanie techniczne i graficzne: Wiesław Gardas
Nakład 200 000
Druk: Szczecińskie Zakłady Graficzne
nr indeksu 396931
nr zamówienia 1651/11.10 N-9
Pozycja wydrukowana na papierze niepełnowartościowym spoza puli RSW.
Cena zł 65,—

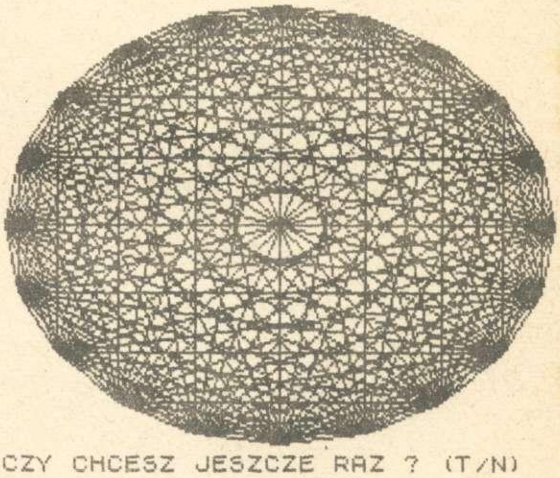
WIELOKĄTY

Żeby szybko narysować wielobok wpisany w owal ze wszystkimi jego przekątnymi, można posłużyć się tym programem. Dla większej ilości wierzchołków (ponad dwadzieścia kilka) wynikiem końcowym jest wypełnienie całego owalu. W takich przypadkach można przerwać rysowanie w dowolnym miejscu klawiszem BREAK.

```
   10 REM ************************
   20 REM ****** WIELOKATY ******
   30 REM ************************
   40 DIM W(200)
   50 INPUT "DLUGOSC PROMIENIA (m
ax. 84) ";R: IF R<1 OR R>84 THEN
 GO TO 50
   60 INPUT "ILOSC WIERZCHOLKOW (
max. 100) ";V: IF V<0 OR V>100 T
HEN GO TO 60
   70 CLS : LET X=127: LET Y=87
   80 LET X0=X+1.5*R: LET Y0=Y: L
ET T=2*PI/V: LET N=V-1
   90 FOR I=1 TO N
  100 LET W(2*I-1)=1.5*R*COS (T*I
)+X
  110 LET W(2*I)=R*SIN (T*I)+Y
  120 PLOT W(2*I-1),W(2*I)
  130 NEXT I
  140 FOR I=1 TO V-1
  150 FOR J=1 TO N
  160 LET XN=W(2*J-1): LET YN=W(2
*J)
  170 PLOT X0,Y0: DRAW XN-X0,YN-Y
0
  180 NEXT J
  190 LET X0=XN: LET Y0=YN: LET N
=N-1
  200 NEXT I
  210 PRINT #0;"CZY CHCESZ JESZCZ
E RAZ ? (T/N)"
  220 LET A$=INKEY$: IF A$="" THE
N GO TO 220
  230 IF A$="T" OR A$="t" THEN RU
N

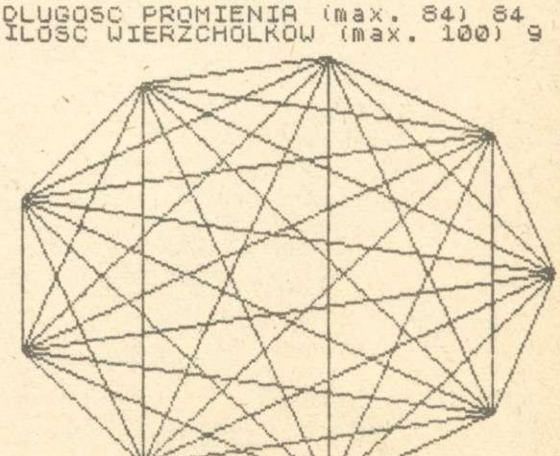
RUN

DLUGOSC PROMIENIA (max. 84) 84
ILOSC WIERZCHOLKOW (max. 100) 20
```



```
CZY CHCESZ JESZCZE RAZ ? (T/N)

DLUGOSC PROMIENIA (max. 84) 84
ILOSC WIERZCHOLKOW (max. 100) 9
```



```
CZY CHCESZ JESZCZE RAZ ? (T/N)
```